Docket No. 218943US2RD

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

IN RE APPLICATION OF: Mikio HASHIMOTO, et al. GAU:

SERIAL NO: NEW APPLICATION EXAMINER:

FILED: HEREWITH

FOR: MICROPROCESSOR USING ASYNCHRONOUS PUBLIC KEY DECRYPTION PROCESSING

REQUEST FOR PRIORITY

ASSISTANT COMMISSIONER FOR PATENTS
WASHINGTON, D.C. 20231

SIR:

☐ Full benefit of the filing date of U.S. Application Serial Number , filed , is claimed pursuant to the provisions of **35 U.S.C. §120**.

☐ Full benefit of the filing date of U.S. Provisional Application Serial Number , filed , is claimed pursuant to the provisions of **35 U.S.C. §119(e)**.

☒ Applicants claim any right to priority from any earlier filed applications to which they may be entitled pursuant to the provisions of **35 U.S.C. §119**, as noted below.

In the matter of the above-identified application for patent, notice is hereby given that the applicants claim as priority:

COUNTRY	APPLICATION NUMBER	MONTH/DAY/YEAR
JAPAN	2001-024480	January 31, 2001

Certified copies of the corresponding Convention Application(s)

☒ are submitted herewith

☐ will be submitted prior to payment of the Final Fee

☐ were filed in prior application Serial No. filed

☐ were submitted to the International Bureau in PCT Application Number
Receipt of the certified copies by the International Bureau in a timely manner under PCT Rule 17.1(a) has been acknowledged as evidenced by the attached PCT/IB/304.

☐ (A) Application Serial No.(s) were filed in prior application Serial No. filed ; and

☐ (B) Application Serial No.(s)

- ☐ are submitted herewith
- ☐ will be submitted prior to payment of the Final Fee

Respectfully Submitted,

OBLON, SPIVAK, McCLELLAND,
MAIER & NEUSTADT, P.C.

Marvin J. Spivak
Registration No. 24,913

James D. Hamilton
Registration No. 28,421

22850
Tel. (703) 413-3000
Fax. (703) 413-2220
(OSMMN 10/98)

日　本　国　特　許　庁
JAPAN PATENT OFFICE

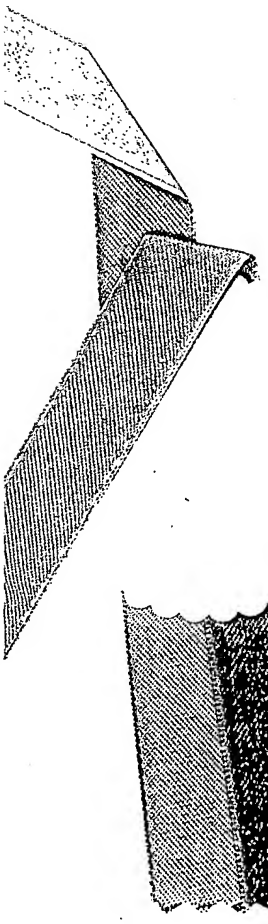
別紙添付の書類に記載されている事項は下記の出願書類に記載されている事項と同一であることを証明する。

This is to certify that the annexed is a true copy of the following application as filed with this Office

出願年月日
Date of Application: ２００１年　１月３１日

出願番号
Application Number: 特願２００１－０２４４８０

出願人
Applicant(s): 株式会社東芝



CERTIFIED COPY OF
PRIORITY DOCUMENT

２００１年１２月２１日

特許庁長官
Commissioner,
Japan Patent Office

及川耕造

出証番号　出証特２００１－３１１０９２１

【書類名】　特許願
【整理番号】　A000100250
【提出日】　平成13年 1月31日
【あて先】　特許庁長官　殿
【国際特許分類】　G06F 15/00
【発明の名称】　マイクロプロセッサ
【請求項の数】　6
【発明者】
【住所又は居所】　神奈川県川崎市幸区小向東芝町１番地　株式会社東芝研究開発センター内
【氏名】　橋本　幹生
【発明者】
【住所又は居所】　神奈川県川崎市幸区小向東芝町１番地　株式会社東芝研究開発センター内
【氏名】　白川　健治
【発明者】
【住所又は居所】　神奈川県川崎市幸区小向東芝町１番地　株式会社東芝研究開発センター内
【氏名】　寺本　圭一
【発明者】
【住所又は居所】　神奈川県川崎市幸区小向東芝町１番地　株式会社東芝研究開発センター内
【氏名】　藤本　謙作
【発明者】
【住所又は居所】　神奈川県川崎市幸区小向東芝町１番地　株式会社東芝研究開発センター内
【氏名】　尾崎　哲
【特許出願人】
【識別番号】　000003078

【氏名又は名称】　株式会社　東芝

【代理人】

【識別番号】　100058479

【弁理士】

【氏名又は名称】　鈴江　武彦

【電話番号】　03-3502-3181

【選任した代理人】

【識別番号】　100084618

【弁理士】

【氏名又は名称】　村松　貞男

【選任した代理人】

【識別番号】　100068814

【弁理士】

【氏名又は名称】　坪井　淳

【選任した代理人】

【識別番号】　100092196

【弁理士】

【氏名又は名称】　橋本　良郎

【選任した代理人】

【識別番号】　100091351

【弁理士】

【氏名又は名称】　河野　哲

【選任した代理人】

【識別番号】　100088683

【弁理士】

【氏名又は名称】　中村　誠

【選任した代理人】

【識別番号】　100070437

【弁理士】

【氏名又は名称】　河井　将次

【手数料の表示】

【予納台帳番号】　011567

【納付金額】　21,000円

【提出物件の目録】

【物件名】　明細書　1

【物件名】　図面　1

【物件名】　要約書　1

【プルーフの要否】　要

【書類名】　明細書

【発明の名称】　マイクロプロセッサ

【特許請求の範囲】

【請求項１】

外部へ読み出すことのできない固有の秘密鍵を内部に保持した１チップまたは１パッケージのマイクロプロセッサであって、

あらかじめ前記秘密鍵に対応する公開鍵によって暗号化された命令鍵をマイクロプロセッサ外部の記憶手段から読み出す第１の読み出し手段と、

前記第１の読み出し手段で読み出した前記命令鍵を、前記秘密鍵を用いて復号化する第１の復号化手段と、

プログラムを識別するプログラム識別子対応に、プログラムを復号化する命令鍵を保持するテーブルと、

前記テーブルに前記プログラム識別子対応に命令鍵を登録する登録手段と、

登録の完了を命令実行部に通知する通知手段と、

実行中のプログラムのプログラム識別子を保持するプログラム識別子保持手段と、

外部の記憶手段の所定のアドレスの内容を前記プログラム識別子保持手段によって指示される前記テーブルに格納された命令鍵によって復号化する第２の復号化手段と、

外部の記憶手段から読み込んだ命令および前記第２の復号化手段により復号化された命令を実行する命令実行手段と、

前記第１の復号化手段に命令鍵を復号化させるステップと、該命令において指定されたプログラム識別子と前記命令鍵とを対応づけて前記テーブルに登録するステップと、登録の完了を命令実行部に通知するステップとを実行する第１の命令実行手段と、

前記プログラム識別子保持手段に、該命令において指定されるプログラム識別子を設定するステップと、該命令において指定される外部の記憶手段のアドレスにプログラム実行制御を移すステップとを実行する第２の命令実行手段とを備えたことを特徴とするマイクロプロセッサ。

【請求項２】

外部へ読み出すことのできない固有の秘密鍵を内部に保持した１チップまたは１パッケージのマイクロプロセッサであって、

あらかじめ前記秘密鍵に対応する公開鍵によって暗号化された実行コード暗号化鍵をマイクロプロセッサ外部の記憶手段から読み出す第１の読み出し手段と、

前記第１の読み出し手段で読み出した前記実行コード暗号化鍵を、前記秘密鍵を用いて復号化する第１の復号化手段と、

前記復号化された命令鍵を、プログラム識別子に対応づけて保持する第１のテーブルと、

前記テーブルに前記プログラム識別子対応に命令鍵を登録する登録手段と、

実行中のプログラムのプログラム識別子を保持するプログラム識別子保持手段と、

前記プログラム識別子をプログラム読出し要求に付随して出力するプログラム実行手段と、

前記プログラム読出し要求に付随する前記プログラム識別子によって該キャッシュラインの暗号鍵が選択され、復号化が行われる第２の復号化手段と、

前記第２の復号化手段の復号化結果と前記読出し要求に付随するプログラム識別子と対応付けてキャッシュライン毎に保持する手段と、

前記キャッシュメモリのキャッシュラインに保持されるプログラム識別子とプログラム実行手段から出力されるプログラム読出し要求のプログラム識別子とが一致したときのみ読出しを許可するキャッシュメモリと、

前記登録手段が第１のテーブルの第１のプログラム識別子に対応する命令鍵を書き換えるとき、前記第１のプログラム識別子を保持する前記キャッシュメモリ上のキャッシュラインをフラッシュする鍵管理手段とを備えたことを特徴とするマイクロプロセッサ。

【請求項３】

外部へ読み出すことのできない固有の秘密鍵を内部に保持した１チップまたは１パッケージのマイクロプロセッサであって、

あらかじめ前記秘密鍵に対応する公開鍵によって暗号化された命令鍵をマイク

ロプロセッサ外部の記憶手段から読み出す第１の読み出し手段と、

前記第１の読み出し手段で読み出した前記命令鍵を、前記秘密鍵を用いて復号化する第１の復号化手段と、

外部の記憶手段におかれている、暗号単位間の連鎖の有無を定義する連鎖情報を読み込む連鎖情報取得手段と、

プログラムを識別するプログラム識別子対応に、前記復号化された命令鍵を保持するテーブルと、

前記連鎖情報取得手段が読み込んだ連鎖情報に基づいて、外部の記憶手段の所定のアドレスにおかれた複数の暗号単位を前記命令鍵によって復号化する第２の復号化手段とを備えたことを特徴とするマイクロプロセッサ。

【請求項４】

外部へ読み出すことのできない固有の秘密鍵を内部に保持した１チップまたは１パッケージのマイクロプロセッサであって、

あらかじめ前記秘密鍵に対応する公開鍵によって暗号化された命令鍵をマイクロプロセッサ外部の記憶手段から読み出す第１の読み出し手段と、

前記第１の読み出し手段で読み出した前記命令鍵を、前記秘密鍵を用いて復号化する第１の復号化手段と、

実行中のプログラムのプログラム識別子を保持するプログラム識別子保持手段と、

登録される鍵の値についてテーブル内での一意性を保証する鍵登録機構を持つ、復号鍵を保持する第１のテーブルと、

プログラム識別子によって参照され、前記第１のテーブルへの参照を保持する第２のテーブルと、

キャッシュメモリ参照において、前記プログラム識別子に基づいて前記第２のテーブルを参照し、前記第１のテーブルへの参照に変換を行い、前記第１のテーブルへの参照情報をキャッシュタグとするタグ変換手段と、

キャッシュメモリと、

キャッシュメモリと外部メモリとのデータ読み書きにおいて、キャッシュメモリから提示されたタグ値に基づいて、前記第１のテーブルから対応する前記復号

鍵を取得し、データの暗号化もしくは復号化を行う暗号手段とを備えたことを特徴とするマイクロプロセッサ。

【請求項５】

外部へ読み出すことのできない固有の第１の秘密鍵および第２の秘密鍵を内部に保持した１チップまたは１パッケージのマイクロプロセッサであって、

あらかじめ前記第１の秘密鍵に対応する公開鍵によって一体として暗号化された命令鍵および永続化許可フラグをマイクロプロセッサ外部の記憶手段から読み出す第１の読み出し手段と、

前記第１の読み出し手段で読み出した前記命令鍵および永続化許可フラグを、前記秘密鍵を用いて復号化する第１の復号化手段と、

プログラムを識別するプログラム識別子対応に、前記命令鍵および前記永続化許可フラグおよび共通鍵を保持するテーブルと、

実行中のプログラムのプログラム識別子を保持するプログラム識別子保持手段と、

前記プログラム識別子対応に、プログラムの実行中断時のレジスタ値を前記共通鍵で暗号化してレジスタ情報として外部の記憶手段に保存するレジスタ値保存手段と、

前記プログラム識別子対応に、保存された前記レジスタ情報を前記共通鍵により復号化してレジスタに読み込むコンテキスト復帰手段と、

該命令において指定されるプログラム識別子に対応する前記永続化許可フラグの値が所定の値をとる場合のみ、前記プログラム識別子に対応する前記共通鍵を前記プロセッサの第２の秘密鍵により暗号化して外部の記憶手段に書き出すステップを実行する所定の命令実行手段とを備えたことを特徴とするマイクロプロセッサ。

【請求項６】

外部へ読み出すことのできない固有の第１の秘密鍵を内部に保持した１チップまたは１パッケージのマイクロプロセッサであって、

あらかじめ前記秘密鍵に対応する公開鍵によって一体として暗号化された命令鍵およびフィードバック鍵をマイクロプロセッサ外部の記憶手段から読み出す第

１の読み出し手段と、

前記第１の読み出し手段で読み出した前記命令鍵およびフィードバック鍵を、前記秘密鍵を用いて復号化する第１の復号化手段と、

プログラムを識別する識別子対応に、前記命令鍵およびフィードバック鍵を保持するテーブルと、

実行中のプログラムのプログラム識別子を保持するプログラム識別子保持手段と、

前記プログラム識別子対応に、プログラム実行中断時のレジスタ値を前記共通鍵で暗号化してレジスタ情報として外部の記憶手段に保存するレジスタ値保存手段と、

該命令において指定される所定のプログラム識別子に対応する前記共通鍵を前記フィードバック鍵により暗号化して外部の記憶手段に書き出すステップを実行する所定の命令実行手段とを備えたことを特徴とするマイクロプロセッサ。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】

本発明は、マルチタスクのプログラム実行環境下で、実行コードや処理対象であるデータの不正な改変を防止することのできるマイクロプロセッサ、これを用いたマルチタスク実行方法、およびマルチレッド実行方法に関する。

【０００２】

【従来の技術】

オープンシステムが広く普及している。オープンシステムでは、ＰＣなどの一般ユーザ向けコンピュータのハードウェアやＯＳと呼ばれるシステムプログラムの情報が開示されており、システムプログラムを変更して望んだ改良を加えることができる。

【０００３】

このような環境で、アプリケーションプログラムが扱う情報の著作権保護や、プログラム自体の著作権保護を保証するには、システムのＯＳがアプリケーションに対して敵対的動作をとることを前提としたハードウェアが必要となる。この

ようなハードウェア、特にマイクロプロセッサの提案が行われている（橋本他、特願２０００－３５３９８，Ｌｉｅ他，“Ａｒｃｈｉｔｅｃｔｕｒａｌ Ｓｕｐｐｏｒｔ ｆｏｒ Ｃｏｐｙ ａｎｄ Ｔａｍｐｅｒ Ｒｅｓｉｓｔａｎｔ Ｓｏｆｔｗａｒｅ”；Ｃｏｍｐｕｔｅｒ Ａｒｃｈｉｔｅｃｔｕｒｅ Ｎｅｗｓ ２８（５）：ｐ１６８－）。このようなマイクロプロセッサは、マルチタスク環境でプログラムとそれが扱う情報の覗き見、改変から守るためにそれら情報を暗号化する機能を備えている。以下、このようなマイクロプロセッサを耐タンパマイクロプロセッサと呼ぶ。

【０００４】

アプリケーションプログラムを構成する３要素（プログラム、コンテキスト、データ）を、プロセッサ外部において暗号化して保護する機構については橋本らによる文献で開示されている。この従来例においては、外部メモリ上で暗号化された３要素の情報が、プロセッサ内部のキャッシュメモリおよびプロセッサレジスタでは、復号化された状態で処理される。

【０００５】

情報を厳密に守るためには、プログラムＡが動作することによってプロセッサ内部に読み込まれ、平文状態で保持される情報が、ＯＳや別のプログラムＢによって読み出されることを防ぐ機構が必要となる。

【０００６】

このとき、第１の前提は、プログラムの識別は、プログラムの命令が暗号化された暗号鍵によって識別されることである。例えば、あるプログラムＡがＫａという鍵で暗号化されているとき、平文のプログラムＢや別の暗号鍵Ｋｃで暗号化されたプログラムＣからは、プログラムＡの情報３要素（プログラム、コンテキスト、データ）を読み出すことはできないことが、耐タンパマイクロプロセッサに対する基本的な要求条件である。プログラムＡについては、プログラム提供者のみが知るその暗号鍵Ｋａによって暗号化したプログラムを配布し、Ｋａを秘密としておくことで、他のプログラム作成者のプログラムと、プログラムＡとを区別することができる。以下、プログラムＡの命令の暗号鍵Ｋａを、プログラムＡの「命令鍵」と呼ぶこととする。

【０００７】

この要求条件を実現する機構として、従来から存在するプロセッサのタグメモリによるアクセス制御機構を利用することが効率的である。これは白川らによる特願２０００－３３３６３５に開示されている。

【０００８】

だが、従来のシステムでは、タグの管理はＯＳに委ねられていた。ところが、耐タンパプロセッサのシステムでは、タグの管理はユーザによる改変操作が可能なソフトウェアのＯＳではなく、プロセッサ内部にあらかじめ組み込まれていてユーザによる改変が困難なハードウェアまたはソフトウェアの機構によって行われる必要がある。

【０００９】

このような機構はプロセッサ内部に組み込まれるため、できる限り単純にすることでプロセッサを安価にし、できる限り処理を単純にしてタグ管理によるオーバヘッドを小さくすることが望ましい。また、有限の資源であるタグを再利用するために必要となる、鍵値の書換え操作の前後に置いても前記の要求条件に参照して違反や矛盾が生じることなくタグ管理が行われなければならない。

【００１０】

また、オーバヘッドについてはシステム全体の処理速度の観点と、プロセスの起動または切り替え時の応答時間の観点とがある。この両者を小さくすることも求められる。さらに、従来例では命令を復号化する鍵を取得するための公開鍵復号化処理と、取得した鍵を使用してのプログラム復号化と実行がひとつの命令で実行されていた。一般のプロセッサでは命令の実行中に例外をうけつけることはできない。例えば、１０２４ビットの公開鍵復号化処理は、現時点で最も高速なハードウェアでも１ｍｓｅｃ以上の時間を必要とする。これは現在の一般的なリアルタイム処理の応答性能の要求である数十μｓｅｃと比較して非常に長く、システムのリアルタイム応答性能を低下させる原因となる。

【００１１】

【発明が解決しようとする課題】

本発明は、上記事情を考慮してなされたもので、キャッシュメモリ管理手順に

おいて、初期登録における公開鍵処理時間による応答不感時間を短縮して、リアルタイム処理性能を向上させることのできるマイクロプロセッサを提供することを目的とする。

【００１２】

また、本発明は、キャッシュメモリのアクセス制御に使われるタグの安全な管理を可能にしたマイクロプロセッサを提供することを目的とする。

【００１３】

また、本発明は、連鎖によってプログラムの暗号鍵を推定されにくくすると同時に効率的な命令実行を実現するプログラム復号化手段を持つマイクロプロセッサを提供することを目的とする。

【００１４】

また、本発明は、複数のプロセスが同一の暗号鍵によって暗号化されているデータを共有をするときの処理効率を向上させることのできるマイクロプロセッサを提供することを目的とする。

【００１５】

また、本発明は、安全なプログラム利用の利便性と秘密の安全性という、相反する要素を、プログラム提供者の意志にしたがって実現するコンテキスト保存手段を持つマイクロプロセッサを提供することを目的とする。

【００１６】

【課題を解決するための手段】

本発明は、外部へ読み出すことのできない固有の秘密鍵を内部に保持した１チップまたは１パッケージのマイクロプロセッサであって、あらかじめ前記秘密鍵に対応する公開鍵によって暗号化された命令鍵をマイクロプロセッサ外部の記憶手段から読み出す第１の読み出し手段と、前記第１の読み出し手段で読み出した前記命令鍵を、前記秘密鍵を用いて復号化する第１の復号化手段と、プログラムを識別するプログラム識別子対応に、プログラムを復号化する命令鍵を保持するテーブルと、前記テーブルに前記プログラム識別子対応に命令鍵を登録する登録手段と、登録の完了を命令実行部に通知する通知手段と、実行中のプログラムのプログラム識別子を保持するプログラム識別子保持手段と、外部の記憶手段の所

定のアドレスの内容を前記プログラム識別子保持手段によって指示される前記テーブルに格納された命令鍵によって復号化する第２の復号化手段と、外部の記憶手段から読み込んだ命令および前記第２の復号化手段により復号化された命令を実行する命令実行手段と、前記第１の復号化手段に命令鍵を復号化させるステップと、該命令において指定されたプログラム識別子と前記命令鍵とを対応づけて前記テーブルに登録するステップと、登録の完了を命令実行部に通知するステップとを実行する第１の命令実行手段と、前記プログラム識別子保持手段に、該命令において指定されるプログラム識別子を設定するステップと、該命令において指定される外部の記憶手段のアドレスにプログラム実行制御を移すステップとを実行する第２の命令実行手段とを備えたことを特徴とする。

【００１７】

本発明は、外部へ読み出すことのできない固有の秘密鍵を内部に保持した１チップまたは１パッケージのマイクロプロセッサであって、あらかじめ前記秘密鍵に対応する公開鍵によって暗号化された実行コード暗号化鍵をマイクロプロセッサ外部の記憶手段から読み出す第１の読み出し手段と、前記第１の読み出し手段で読み出した前記実行コード暗号化鍵を、前記秘密鍵を用いて復号化する第１の復号化手段と、前記復号化された命令鍵を、プログラム識別子に対応づけて保持する第１のテーブルと、前記テーブルに前記プログラム識別子対応に命令鍵を登録する登録手段と、実行中のプログラムのプログラム識別子を保持するプログラム識別子保持手段と、前記プログラム識別子をプログラム読出し要求に付随して出力するプログラム実行手段と、記プログラム読出し要求に付随する前記プログラム識別子によって該キャッシュラインの暗号鍵が選択され、復号化が行われる第２の復号化手段と、前記第２の復号化手段の復号化結果と前記読出し要求に付随するプログラム識別子と対応付けてキャッシュライン毎に保持する手段と、前記キャッシュメモリのキャッシュラインに保持されるプログラム識別子とプログラム実行手段から出力されるプログラム読出し要求のプログラム識別子とが一致したときのみ読出しを許可するキャッシュメモリと、前記登録手段が第１のテーブルの第１のプログラム識別子に対応する命令鍵を書き換えるとき、前記第１のプログラム識別子を保持する前記キャッシュメモリ上のキャッシュラインをフラ

ッシュする鍵管理手段とを備えたことを特徴とする。

【００１８】

本発明は、外部へ読み出すことのできない固有の秘密鍵を内部に保持した１チップまたは１パッケージのマイクロプロセッサであって、あらかじめ前記秘密鍵に対応する公開鍵によって暗号化された命令鍵をマイクロプロセッサ外部の記憶手段から読み出す第１の読み出し手段と、前記第１の読み出し手段で読み出した前記命令鍵を、前記秘密鍵を用いて復号化する第１の復号化手段と、外部の記憶手段におかれている、暗号単位間の連鎖の有無を定義する連鎖情報を読み込む連鎖情報取得手段と、プログラムを識別するプログラム識別子対応に、前記復号化された命令鍵を保持するテーブルと、前記連鎖情報取得手段が読み込んだ連鎖情報に基づいて、外部の記憶手段の所定のアドレスにおかれた複数の暗号単位を前記命令鍵によって復号化する第２の復号化手段とを備えたことを特徴とする。

【００１９】

本発明は、外部へ読み出すことのできない固有の秘密鍵を内部に保持した１チップまたは１パッケージのマイクロプロセッサであって、あらかじめ前記秘密鍵に対応する公開鍵によって暗号化された命令鍵をマイクロプロセッサ外部の記憶手段から読み出す第１の読み出し手段と、前記第１の読み出し手段で読み出した前記命令鍵を、前記秘密鍵を用いて復号化する第１の復号化手段と、実行中のプログラムのプログラム識別子を保持するプログラム識別子保持手段と、登録される鍵の値についてテーブル内での一意性を保証する鍵登録機構を持つ、復号鍵を保持する第１のテーブルと、プログラム識別子によって参照され、前記第１のテーブルへの参照を保持する第２のテーブルと、キャッシュメモリ参照において、前記プログラム識別子に基づいて前記第２のテーブルを参照し、前記第１のテーブルへの参照に変換を行い、前記第１のテーブルへの参照情報をキャッシュタグとするタグ変換手段と、キャッシュメモリと、キャッシュメモリと外部メモリとのデータ読み書きにおいて、キャッシュメモリから提示されたタグ値に基づいて、前記第１のテーブルから対応する前記復号鍵を取得し、データの暗号化もしくは復号化を行う暗号手段とを備えたことを特徴とする。

【００２０】

本発明は、外部へ読み出すことのできない固有の第１の秘密鍵および第２の秘密鍵を内部に保持した１チップまたは１パッケージのマイクロプロセッサであって、あらかじめ前記第１の秘密鍵に対応する公開鍵によって一体として暗号化された命令鍵および永続化許可フラグをマイクロプロセッサ外部の記憶手段から読み出す第１の読み出し手段と、前記第１の読み出し手段で読み出した前記命令鍵および永続化許可フラグを、前記秘密鍵を用いて復号化する第１の復号化手段と、プログラムを識別するプログラム識別子対応に、前記命令鍵および前記永続化許可フラグおよび共通鍵を保持するテーブルと、実行中のプログラムのプログラム識別子を保持するプログラム識別子保持手段と、前記プログラム識別子対応に、プログラムの実行中断時のレジスタ値を前記共通鍵で暗号化してレジスタ情報として外部の記憶手段に保存するレジスタ値保存手段と、前記プログラム識別子対応に、保存された前記レジスタ情報を前記共通鍵により復号化してレジスタに読み込むコンテキスト復帰手段と、該命令において指定されるプログラム識別子に対応する前記永続化許可フラグの値が所定の値をとる場合のみ、前記プログラム識別子に対応する前記共通鍵を前記プロセッサの第２の秘密鍵により暗号化して外部の記憶手段に書き出すステップを実行する所定の命令実行手段とを備えたことを特徴とする。

【００２１】

本発明は、外部へ読み出すことのできない固有の第１の秘密鍵を内部に保持した１チップまたは１パッケージのマイクロプロセッサであって、あらかじめ前記秘密鍵に対応する公開鍵によって一体として暗号化された命令鍵およびフィードバック鍵をマイクロプロセッサ外部の記憶手段から読み出す第１の読み出し手段と、前記第１の読み出し手段で読み出した前記命令鍵およびフィードバック鍵を、前記秘密鍵を用いて復号化する第１の復号化手段と、ログラムを識別する識別子対応に、前記命令鍵およびフィードバック鍵を保持するテーブルと、実行中のプログラムのプログラム識別子を保持するプログラム識別子保持手段と、前記プログラム識別子対応に、プログラム実行中断時のレジスタ値を前記共通鍵で暗号化してレジスタ情報として外部の記憶手段に保存するレジスタ値保存手段と、該命令において指定される所定のプログラム識別子に対応する前記共通鍵を前記フ

ィードバック鍵により暗号化して外部の記憶手段に書き出すステップを実行する所定の命令実行手段とを備えたことを特徴とする。

【００２２】

なお、装置に係る本発明は方法に係る発明としても成立し、方法に係る本発明は装置に係る発明としても成立する。

また、装置または方法に係る本発明は、コンピュータに当該発明に相当する手順を実行させるための（あるいはコンピュータを当該発明に相当する手段として機能させるための、あるいはコンピュータに当該発明に相当する機能を実現させるための）プログラムとしても成立し、該プログラムを記録したコンピュータ読取り可能な記録媒体としても成立する。

【００２３】

【発明の実施の形態】

以下、図面を参照しながら発明の実施の形態を説明する。

【００２４】

（第１の実施形態）

第１の実施形態では、プロセッサ内部において平文状態で格納される情報のアクセス制御のための鍵情報（実効鍵識別子）の安全かつ効率的な管理方法について説明する。本実施形態では、鍵情報管理機能はハードウェアによる実装として説明しているが、ユーザによる改変が不可能であるという前提が保証されれば、鍵管理機能がプロセッサ内蔵ＲＯＭに格納されたファームウェアによる実装であってもよい。

【００２５】

はじめに本実施形態における用語を定義する。

【００２６】

「暗号化」は、対称（共通）鍵アルゴリズム、非対称（公開）鍵アルゴリズムの両方を含めて、平文の情報を暗号化することを意味する。

【００２７】

同様に、「復号化」は、対称（共通）鍵アルゴリズム、非対称（公開）鍵アルゴリズムの両方を含めて、暗号化された情報を復号化して平文に戻すことを意味

する。

【００２８】

「暗号鍵」は、上記暗号化、復号化で使われる鍵を暗号化、復号化の両方の場合を含んで用いる。

【００２９】

「共通鍵」は、共通鍵アルゴリズムで暗号化、復号化に共通に使われる鍵を意味する。

【００３０】

「暗号化鍵」は、公開鍵アルゴリズムで暗号化に使われる鍵を意味する。

【００３１】

「復号化鍵」は、公開鍵アルゴリズムで復号化に使われる鍵を意味する。

【００３２】

本実施形態のマイクロプロセッサは、外部から読み出し不能な復号化鍵（秘密鍵以下、Ｋｓと表記）を備える。プロセッサの秘密鍵Ｋｓに対応する暗号化鍵（公開鍵：以下、Ｋｐと表記する）は公開されている。

【００３３】

「暗号化プログラム」は、命令が、あるひとつの共通鍵（ここでは、Ｋｘと表記）によって暗号化されたプログラムを意味する。

【００３４】

あるプログラム作成者が暗号化プログラムを作成するとき、プログラムを暗号化する適切な共通鍵Ｋｘを選び、プログラムの命令を暗号化する（ただし、必ずしもプログラムの全体が暗号化されている必要はない）。

【００３５】

本実施形態のプロセッサにおいて、プログラムの同一性はこの共通鍵Ｋｘによって判定される。この共通鍵Ｋｘを、その暗号化プログラムの命令鍵と呼ぶ。

【００３６】

そして、ターゲットのプロセッサの公開鍵Ｋｐで命令鍵Ｋｘを暗号化した値（Ｅ＿Ｋｐ［Ｋｘ］と表記）を、配布鍵と呼ぶ。プログラム作成者は、命令鍵で暗号化されたプログラム本体と、配布鍵をターゲットのシステムに配布する。

【００３７】

また、コンテキスト情報の暗号処理に使われる暗号鍵をコンテキスト鍵、データの暗号処理に使われる暗号鍵をデータ鍵と呼ぶ。

【００３８】

キャッシュメモリとメインメモリとの間での情報の復号化・暗号化では、命令、データ、コンテキストで何種類もの暗号鍵が使われる。ある特定のキャッシュラインの暗号処理に使われる共通鍵を実効鍵と呼ぶ。

【００３９】

本実施形態での暗号アルゴリズムの扱いでは、対称（共通）鍵アルゴリズム、非対称（公開）鍵アルゴリズムの区別は重要だが、対称（共通）鍵アルゴリズム、非対称（公開）鍵アルゴリズムの中での方式、例えば、ＤＥＳ，Ｒｉａｎｄａｌ，ＲＳＡ，Ｅｌｇａｍａｌなどの暗号方式の違いは重要ではない。したがって、対称（共通）鍵アルゴリズムではＤＥＳ，Ｒｉａｎｄａｌなどが、非対称（公開）鍵アルゴリズムではＲＳＡ，Ｅｌｇａｍａｌなどのいずれを用いてもよい。

【００４０】

プログラムは、命令と静的データで構成される。暗号化プログラムでは、命令は暗号化されているが、静的データは暗号化されていても平文でもどちらでもよい。

【００４１】

プロセスとは、プログラムがターゲットシステムのＯＳの管理のマルチタスク環境下で、実行中の、または実行を中断されている状態をあらわす。プロセスは、命令、実行状態、データからなる。データには静的データと動的データとがある。プロセスの実行は、プログラムの命令が（暗号化プログラムの場合は復号化されて）プロセッサに読み込まれ、実行されることで行われる。プロセスの実行は、割り込みなどの原因で中断されることがある。ＯＳはそのときプロセスがプロセッサ内部に保持する状態、一般にはレジスタ情報を実行状態（コンテキスト情報とも呼ばれる）としてプロセッサ外部のメモリに保存する。割り込みサービスや別のプログラムを実行した後、プロセスの実行を再開するときはメモリに保存した実行状態を再度プロセッサ内部に読み込んで実行を再開する。

【００４２】

ひとつのプログラムから複数のプロセスが作られ、同時に実行されることがある。暗号化プログラムの場合、これらの共通の起源をもつプロセスの間で復化された命令を共有することが可能である。

【００４３】

また、本発明ではキャッシュメモリ上におけるプロセスの秘密情報のアクセス制御をキャッシュタグによって行っているが、このアクセス制御機構と、従来から存在する仮想記憶機構におけるプロセスのアクセス制御機構（ＵＮＩＸカーネル）とは併用が可能である。本発明が新たに導入する前者の機構は、アプリケーションプログラムの作成者が意図する秘密保護が正しくプロセッサ上で実現されることを保証する機構である。具体的にはプログラム１の命令が、他のプログラムから読み取られることなどを防ぐものである。

【００４４】

一方、後者はターゲットシステムのＯＳのシステム管理意図を正しく実現するために用いられる。具体的には、あるプロセス１のメモリアクセスを、他のプロセスから隔離し、互いの動作の干渉を防ぐなどを目的とする。

【００４５】

以下ではこの２種類の機構をそれぞれプロセスの暗号化属性保護、記憶領域保護と呼ぶこととする。本実施形態では、これらの機構はともにキャッシュメモリ上のタグと呼ばれる属性を利用して効率的に実装される。後述のように、タグについては暗号化属性保護、記憶領域保護の領域が設けられている。この両者を区別するとき、暗号化属性（保護）タグ、記憶領域（保護）タグと呼ぶこととする。あるキャッシュデータが暗号処理をうける際の暗号鍵は暗号化属性タグによって決定される。このときの暗号鍵を暗号化属性タグに対する実効鍵と呼ぶ。

【００４６】

なお、第１の実施形態においては、プロセッサの鍵管理機構を説明するため、暗号化された命令の実行のみを説明する。アプリケーションプログラムの残りの要素である実効状態やデータの暗号化は第２の実施形態以降で説明する。

【００４７】

（プログラミングモデル）

はじめに本実施形態のプロセッサにおいて、暗号化プログラムの命令の実行手順をおおまかに説明し、その後、プログラムの秘密を守るプロセッサ内部の機構について詳細に説明する。

【００４８】

図１に、本実施形態に係るマイクロプロセッサの全体構成例を示す。

【００４９】

図１において、１０１はプロセッサ全体を、１０２は内部データバスを、１０３は外部バスインタフェースをそれぞれ表す。外部バスインタフェース１０３は、外部メモリ（図示せず）との間でデータをやり取りする。なお、図１ではアドレスバスは省略されているが、実際にはアドレスバスがある。２０１はプロセッサコアであり、命令キャッシュ３０１、データキャッシュ４０１、鍵管理部７０１に接続されている。

【００５０】

第１の実施形態では、命令のみが暗号化されており、データの暗号化を行わないので、命令復号化機能５０１は命令キャッシュ対応のみ設けられている。プロセッサコア２０１からは、アドレスとその他の属性からなるキャッシュタグが出力され、タグが一致すると、プロセッサコア２０１に命令が出力される。

【００５１】

［平文プログラムの実行］

本実施形態のプロセッサは、メインメモリにおかれた、平文の命令と暗号化された命令が混在したプログラムを実行可能である。

【００５２】

図２で１００１はメモリ空間全体を表す。プログラムはメインメモリ上の１００２～１００４の領域に置かれる。１００２，１００４は平文の領域であり、１００３は暗号化された領域である。領域１００５は１００３を復号化するための鍵情報が格納されている。

【００５３】

［暗号化済み命令の実行］

次に、暗号化された命令（以下、暗号化済み命令と呼ぶ）を実行する場合を説明する。本実施形態のプロセッサには、平文命令の実行と暗号化済み命令の実行の２つの状態があり、この状態を制御する２種類の命令が備えられている。ひとつは平文命令の実行から暗号化済み命令の実行に移行する暗号実行開始命令であり、もうひとつはその逆の平文復帰命令である。これら２つに加えて命令鍵を登録するための鍵登録命令がある。

【００５４】

［鍵登録命令］

鍵登録命令は、次のニーモニックｓｅｔｋｅｙで表記され、ふたつのオペランドをとる。

ｓｅｔｋｅｙ　Ｒａ，　Ｒｂ

Ｒａは任意のレジスタであり、プログラムの配布鍵のアドレスが格納されていることを前提とする。Ｒｂはプロセス識別子を指定する。

【００５５】

ＯＳは暗号化プログラムを実行する前の準備としてそのプログラムの配布鍵アドレスをパラメータとしてｓｅｔｋｅｙ命令を発行する。ｓｅｔｋｅｙ命令を実行したプロセッサは配布鍵を読み取り、それをプロセッサ固有の秘密鍵によって復号化して命令鍵を取り出し、プロセッサ内部に登録する。

【００５６】

次に、登録の内部動作を説明する。

【００５７】

図３に鍵管理部の構成例を示し、図４に鍵登録処理の手順例を示す。

【００５８】

［鍵テーブルの初期化と登録、割り込みによる通知］

ｓｅｔｋｅｙ命令が実行されると、配布鍵のアドレスとプロセス識別子が鍵管理部７０１の状態管理機能７０２に送られる（Ｓ１１０１）。ｓｅｔｋｅｙ命令の実行はこれで完了し、続く命令が実行されるが、登録作業は鍵管理部７０１によって並行して行われ、登録が完了するとプロセッサコアへ割り込み要求として通知される。

【００５９】

鍵管理部７０１での登録作業は大きく２つに分けられる。ひとつは命令鍵の鍵テーブルへの登録であり、もうひとつは命令キャッシュに存在する、新たに登録されるプロセス識別子（＃ｎとする）の暗号化属性タグを持つキャッシュラインを無効化することである。

【００６０】

前者については、状態管理機能７０２がプロセッサコア２０１から受け取った配布鍵のアドレスを配布鍵読み出し機能７０３に送り、内部バス１０２を通じて外部メモリから配布鍵を読み込み、実効鍵復号化機能７０５へと送る（Ｓ１１０２）。実効鍵復号化機能７０５はプロセッサ固有の秘密鍵７０４によって配布鍵を復号化して実効鍵を得る。この復号化は公開鍵アルゴリズムの復号化のため、複数命令サイクルの非常に長いクロック数を必要とする。復号の結果得られた実効鍵は、実効鍵テーブル７０６のプロセス識別子で指定されたエントリに格納される。この作業が終了すると状態管理機能７０２に完了が通知される。

【００６１】

この作業と並行して、状態管理機能７０２は、命令キャッシュ３０１にプロセス識別子＃ｎの属性を持つキャッシュエントリを無効化する（Ｓ１１０２）。無効化が完了すると同様に状態管理機能７０２に通知される。

【００６２】

状態管理機能７０２は、このふたつの作業の両方の完了を確認して（Ｓ１１０３）、実効鍵テーブルの対応するエントリの鍵値を古い値からステップＳ１１０２での復号化の結果得られた値に書き換える。

【００６３】

７０２－１は、プロセス状態管理テーブルであり、プロセス対応に鍵登録の状態（書き換え中または登録済み）を保持するものであり、上記鍵操作の間、操作中のプロセスに対応するエントリの値は書き換え中となり、操作が完了すると登録済みとなる。この値は、プロセッサコアからの読み出し要求によって随時読み出される。

【００６４】

（キャッシュ無効化の効果）

あるプロセス識別子に対して、以前に登録されていたものとは異なる別の実効鍵Ｋｎを登録して、命令を実行するとき、命令キャッシュに以前に登録されていた実効鍵Ｋｏによって復号化されたキャッシュラインが存在していると、新たに実効鍵を登録した暗号化プログラムは、本来はプログラムが実行できるはずのない別の暗号化プログラムの命令を実行できることになり、暗号化プログラムの秘密が保護されないことになる。本発明では、鍵登録において登録する実行鍵の属性を持つキャッシュラインの無効化を行うことにより、暗号化プログラムの秘密保護を厳密に保証できる。

【００６５】

また、鍵登録を構成する公開鍵の復号化処理と、キャッシュラインの無効化という、１命令サイクルで完了しない作業に対して、その実行を１命令で行わずに登録の完了を非同期的に割り込みで通知することにより、登録作業中に他のプログラムを実行したり、別の割り込み要求に応えることを可能として、システムの性能と応答性能を高める効果がある。

【００６６】

この２つの作業が完了すると（Ｓ１１０３）、状態管理機能７０２は、プロセッサコアに鍵登録の完了を割り込みによって通知する。鍵登録状態を取得する命令をもうけ、ＯＳがポーリングによって鍵登録状態を取得してもよい。ＯＳは鍵登録完了割り込みの通知を受けると、暗号化プログラムの実行を開始する（Ｓ１１０５）。

【００６７】

［暗号実行開始命令］

暗号実行開始命令は、次のニーモニックｓｔｒｔｅｎｃで表記され、ふたつのオペランドをとる。

ｓｔｒｔｅｎｃ　Ｒａ，　Ｒｂ

ＲａはレジスタをＲし、制御を移す暗号化命令の先頭を示すアドレスが、Ｒｂは先に登録した実行鍵の識別子がＯＳによってそれぞれ格納されていることを前提とする。この場合、Ｒａには暗号化命令列の先頭アドレスｓｔａｒｔ（図２中

、１００８）が格納され、Ｒｂには＃ｎが指定されている。

【００６８】

この他に制御レジスタとして、後述の暗号連鎖ベクトルの格納アドレスと暗号連鎖の先頭オフセットアドレスを格納するレジスタが存在する。暗号連鎖ベクトルレジスタにはｃｈａｉｎａｄｄｒ（図２中、１０１０）が、暗号連鎖オフセットレジスタにはｓｔａｒｔ（図２中、１００８）が設定されている。

【００６９】

ｓｔｒｔｅｎｃ命令が実行されると、実行鍵識別子＃ｎが実行鍵識別子格納レジスタ２０２に読み込まれ、アドレスｓｔａｒｔから暗号化プログラムの実行が始まる。

【００７０】

［命令の復号化と連鎖］

命令の復号化と連鎖の処理について説明する。

【００７１】

図５に、命令復号化処理部５０１の構成例を示す。

【００７２】

プロセッサコア２０１は、命令を読み込むため、アドレスｓｔａｒｔと記憶領域タグ、暗号化属性タグからなる属性情報を、命令キャッシュ３０１に送る。キャッシュがヒットしない場合、アドレス情報が命令復号化処理部５０１に送られ、外部メモリから暗号化プログラムを読み出し、復号化してキャッシュに読み込む。

【００７３】

アドレス情報処理部５０２は、命令キャッシュ３０１からアドレス情報を受け取る。メモリアドレスは外部バスインタフェース１０３に送られ、対応するデータが外部メモリから読み出される。一方、暗号化属性タグは鍵管理部７０１に送られる。その結果として、実効鍵値が鍵管理部７０１から復号化機能５０３に送られ、復号化された平文データがバッファ５０４に格納される。

【００７４】

ここで、連鎖ベクトルについて説明する。

【００７５】

連鎖ベクトルは、暗号ブロックであるキャッシュラインごとに、暗号化に連鎖を使っているかどうか０／１の１ビット値のつらなりからなるベクトルであり、プロセッサの連鎖ベクトルアドレス格納レジスタで指定された場所に置かれる。先頭のビットは連鎖ベースアドレスレジスタで指定されたキャッシュラインに対応する。あるキャッシュラインに対応するビットが１の場合、そのキャッシュラインがひとつ前のキャッシュラインとの間に連鎖の関係を持っていることを表す。

【００７６】

プログラムの先頭は連鎖を取れないので、先頭のビットは必ず０となる。プログラムが先頭から順次実行されていく場合、命令復号化処理部５０１（連鎖計算機能５０５）では順次連鎖が計算され、復号化が行える。

【００７７】

だが、プログラムには分岐やエントリポイントがある。プログラムのエントリポイントや、大域的な分岐点も０とすることが望ましい。もしあるプログラムの先頭を除いて全ての連鎖属性が１となっていた場合、プログラムの最後に対応する部分への分岐を行って、そのキャッシュラインを復号化するには、プログラムの先頭からの連鎖を全て計算しなければならないことになり、大きなオーバヘッドとなってしまう。

【００７８】

暗号化プログラムのキャッシュメモリへの読み込みが、連鎖の途中から開始された場合、復号化処理機能では、連鎖属性が０となるまで前のキャッシュラインを読み込み、順次連鎖の値のみを計算して、読み込んだデータは捨て、最終的に目的のキャッシュラインを復号化して命令キャッシュへと送る。

【００７９】

本実施形態では、連鎖ベクトルとそれに対応する復号化処理機能を設けたことにより、連鎖による暗号強度の向上と、分岐における処理の効率化の両方を達成している。

【００８０】

［命令キャッシュへの読み込み］

復号化されたキャッシュラインが命令キャッシュのあるエントリに格納されると、そのタグ領域に実効鍵識別子を含むタグ情報が書き込まれる。もしキャッシュがフラッシュされないうちにそのプロセスが再度同じアドレスをアクセスすれば、キャッシュがヒットしてキャッシュ上にある平文データが読み出される。

【００８１】

［鍵属性不一致の場合の処理］

さて、本実施形態では、キャッシュタグのアドレスが要求したアドレスと一致しても、実行中のプログラムのプロセス識別子と、キャッシュタグの実効鍵識別子が異なる場合はキャッシュヒットとはみなさない。代わりに、現在有効な実効鍵で復号化されたキャッシュラインが新たに読み込まれ、実行される。

【００８２】

ここでは、命令キャッシュを対象としているので、書き込みは行われないが、第２の実施形態以降での、データキャッシュで同様の不一致が起こった場合、もしそのキャッシュラインに書き込みが行われてｄｉｒｔｙ　ｆｌａｇがセットされていれば、一度キャッシュをフラッシュして、メインメモリにデータが（最初に読み込まれたときの共通鍵で暗号化されて）外部メモリに書き戻され、再び現在有効な共通鍵で復号化されて読み込まれる。

【００８３】

このように、キャッシュアクセスで暗号化属性が一致しないには、現在有効な暗号化属性で再度外部メモリからその命令またはデータを読み出す処理を行い、例外は発生させない。この動作は、実行中のプログラムから見れば、そのキャッシュラインが読み込まれていない場合と、まったく同一である。もしそのキャッシュラインが読み込まれていなければ、外部メモリから、現在有効な暗号化属性によって復号化された命令が読み込まれるからである（それは必ずしも正しく実行される保証はない）。

【００８４】

［平文その他］

なお、命令鍵識別子が０は平文プログラムの実行に使われる。復号化処理機能

から、命令鍵識別子０の復号化鍵値の要求があると、鍵管理部７０１は、予め定められた復号化をしないことを示す値を返す。

【００８５】

また、ある実行鍵識別子＃ｘに対する鍵登録処理が行われている間、プロセッサコアから命令キャッシュのタグ＃ｘを持つキャッシュエントリのアクセスは禁止される。

【００８６】

（第２の実施形態）

第１の実施形態ではシステムの応答性を改善する非同期的な鍵登録処理と、命令のみの暗号化について効率的かつ安全な鍵管理方法を中心に説明した。第２の実施形態では、命令に加えてコンテキストとデータの効率的かつ安全な暗号鍵管理方式を説明する。本実施形態では、第１の実施形態と相違する点を中心にして説明する。

【００８７】

図６に、本実施形態に係るマイクロプロセッサの全体構成例を示す。第１の実施形態の場合に対して、データ暗号化・復号化処理部６０１が追加されている。また、プロセッサコアにデータ暗号化属性レジスタ２０６が追加されている。

【００８８】

図７に、本実施形態における鍵管理部７０１の構成例を示す。命令鍵テーブルに加えてコンテキスト鍵テーブル７０９、データ鍵テーブル７１０が追加されている。

【００８９】

コンテキスト鍵テーブル７０９には、プロセス識別子対応に１つずつの共通鍵を格納する領域が設けられる。プロセス識別子ｉに対しては７０９－ｉが対応する。また、データ鍵テーブル７１０には、プロセス識別子対応に２つずつの共通鍵を格納する領域が設けられる。プロセス識別子ｉに対しては、７１０－ｉ－１と７１０－ｉ－２が対応する。

【００９０】

これらの実効鍵をキャッシュ上で識別するため、暗号化属性タグは２ビット追

加され、プロセス識別子と、２ビットの種別フィールドの組み合わせで実効鍵が決定される。

【００９１】

実効鍵テーブルは、鍵の値を格納するテーブル一般を指示し、命令鍵テーブル、コンテキスト鍵テーブル、データ鍵テーブルは、実効鍵テーブルであって格納する鍵がそれぞれ命令鍵、コンテキスト鍵、データ鍵であるものを指示する。第１の実施形態においては、命令鍵を格納するテーブルを実効鍵テーブルとして説明したが、第２の実施形態以下においては、３種類のテーブルを区別するため、命令鍵テーブル、コンテキスト鍵テーブル、データ鍵テーブルと呼ぶことにする。

【００９２】

（命令鍵登録、命令実行開始）

暗号化プログラムの実行には、まず、命令鍵を命令鍵テーブルに登録する。命令鍵の登録手順は基本的には第１の実施形態の場合と同様であり、上述した鍵再割り当てに伴う命令キャッシュの無効化も行われる。

【００９３】

ただ、プロセス毎のコンテキスト鍵、データ鍵の独立性を保証するために鍵登録時の内部動作が異なっている。命令鍵の登録と同時に、コンテキスト鍵テーブル７０９、データ鍵テーブル７１０についても、プロセス識別子で指定されたエントリが初期化されることである。例えば、プロセス識別子ｉの命令鍵が登録されるときは、コンテキスト鍵テーブル７０９－ｉとデータ鍵テーブル７１０－ｉ－１，７１０－ｉ－２の値が予め定められた平文アクセスに対応する値に初期化される。そして、単に値が初期化されるだけでなく、これらの実効鍵識別子を持つデータキャッシュのラインがフラッシュされる。もし外部メモリに書き戻されていないデータがあればキャッシュから外部メモリにデータが書き戻される。このとき、キャッシュデータの書き戻しに古いコンテキスト鍵、データ鍵の値が必要なので、フラッシュが完了するまでは、古いコンテキスト鍵、データ鍵の値を保持しておく必要がある。

【００９４】

プロセス識別子に対応する命令鍵の登録と、コンテキスト鍵、データ鍵の初期化、そして命令、データキャッシュのフラッシュが完了すると状態制御部はプロセッサコアに登録完了を通知する。

【００９５】

（命令実行開始）

命令実行の開始手順も第１の実施形態と同一である。

【００９６】

異なるのは、例外発生時のコンテキスト情報保護処理と、データ暗号化処理である。

【００９７】

はじめに例外発生時のデータ暗号化処理を説明する。データ暗号化機能は、暗号化プロセスのみが使うことができる。データ暗号化機能は、データ暗号化属性レジスタによって制御されるが、暗号化プログラムの実行開始時にはデータ暗号化属性レジスタの有効ビットがクリアされている。また、暗号化属性を変更する前には、必ず有効ビットを一度クリアしなければならない。有効ビットをクリアすると、対応するタグをもつキャッシュエントリは全てフラッシュされる。

【００９８】

データ暗号化機能を使うには、データ暗号化レジスタの暗号化属性を設定する。暗号化属性は、開始アドレス、対象領域の長さ、データ鍵からなる。これらのパラメータをデータ暗号化属性レジスタに書き込み、有効ビットをセットする。すると、データ鍵のデータ鍵テーブルへの登録と、アドレスーデータ鍵識別子変換表の構築が行われる。

【００９９】

データ鍵の登録は、データ鍵テーブルのプロセス識別子と補助情報によって指定されたエントリにデータ鍵の値を書き込むだけである。この時点では、対応するキャッシュラインは全てフラッシュされているので、あらためてキャッシュのフラッシュを待つ必要はない。

【０１００】

データ鍵の登録と並行して、アドレス－データ鍵識別子変換表の構築が行われ

る。アドレス－データ鍵識別子変換表は、プロセッサ内部のメモリ管理機能の中にある（図示せず）。データ暗号化レジスタの開始アドレス、長さの情報から、アドレス範囲に対してどのデータ鍵識別子が使われるかを対応付けるテーブルが構築される。そして、暗号化対象範囲のキャッシュされたデータは、全てフラッシュされる。

【０１０１】

有効ビットをセットする命令の実行が完了すると、データ暗号化機能が有効な状態となり、データ暗号化属性レジスタに指定された領域のメモリへの読み書きは、全て暗号化されて行われる。

【０１０２】

キャッシュされたメモリへの書き込みは、平文状態で行われるが、対応するデータ鍵識別子のタグが付加される。そのラインがフラッシュされるときはデータ鍵識別子が鍵管理部７０１に送られ、データ鍵テーブル７１０で検索されたデータ鍵が、データ暗号化・復号化処理部６０１に読み込まれ、データは暗号化されてメモリに書き込まれる。

【０１０３】

暗号化により、わずか１ビットのデータの書き換えでも同じキャッシュラインに対応する外部メモリの内容が全面的に書き換わることに注意が必要である。

【０１０４】

読み込みの場合はこの一連の処理の逆が行われる。

【０１０５】

アドレス－データ鍵識別子変換表は、現在実行中のプロセスに対応したもののみがプロセッサコアに保持される。実行するプロセスが切り替えられると、アドレス－データ鍵識別子変換表も再構築される。ただし、プロセスの切り替えではキャッシュのフラッシュを行う必要はない。切り替えの前後で同一のデータ鍵識別子に対応する鍵の値は変わらないからである。

【０１０６】

一方、データ鍵テーブルには、全ての暗号化プロセスのデータ鍵が保持される。これは、フラッシュはキャッシュラインは、現在実行中のプロセスのタグがつ

いたものとは限らないからである。

【０１０７】

上記のデータ鍵管理機能により、プロセスの切り替えや、データ暗号化対象アドレスやデータ暗号鍵の変更があっても、データの安全性、整合性が保障され、同時に効率的なデータの暗号化が実現できる。

【０１０８】

次に、例外発生時のコンテキスト保護処理を説明する。

【０１０９】

暗号化命令の実行中に例外が発生すると、レジスタの内容がレジスタバッファに退避され、下のレジスタは初期化された後に例外ハンドラの処理が開始される。レジスタの退避のときプロセス識別子もレジスタバッファに退避される。レジスタバッファには有効フラグがあり、暗号化プログラムの実行中に例外が発生すると有効フラグが１にセットされる。レジスタバッファは他のプログラムから読み出すことができないので、暗号化プログラムが処理中のレジスタの内容が、例外ハンドラによって読み出されることはない。

【０１１０】

例外ハンドラではレジスタバッファに保存されている中断されたプログラムのレジスタ情報を、次に説明するコンテキスト保存命令によってキャッシュメモリに保存する。

【０１１１】

コンテキスト保存命令（ｓａｖｅｒｅｇ）は次のような形式で表され、一つのオペランドをとる。オペランドｄｅｓｔはレジスタバッファの内容が保存されるアドレスを示す。

ｓａｖｅｒｅｇ　ｄｅｓｔ

ｓａｖｅｒｅｇ命令が発行されるとレジスタバッファの内容がアドレスｄｅｓｔに対応するキャッシュラインに書き込まれる。上述のように通常メモリへのデータ書き込みでは書き込み命令を発行したプロセスのデータ暗号化属性が使われるが、コンテキスト保存命令の書き込みの際には、アクセスに使うタグとして、レジスタバッファに保存されたプロセスのコンテキスト鍵識別子が使われる。

【０１１２】

コンテキストは複数のキャッシュラインに保存され、連鎖ビットが設定される。連鎖により、コンテキストの秘密はより安全になる。

【０１１３】

コンテキスト鍵の扱いは命令鍵、データ鍵とはかなり異なる。あるコンテキスト鍵の値は、乱数生成装置によって決定される。そして、そのプロセスの実行を再開するとき、コンテキストが復号化されるまでプロセッサ内部で保持され、復号化が完了すると捨てられる。

【０１１４】

あるプロセスのコンテキスト鍵は、プロセッサ内部に保持され、外部には読み出せない。物理乱数発生装置などの特性のよい乱数発生装置を使うことにより、同じ鍵が使われる可能性や、鍵の系列を予想することを極めて困難とすることができる。

【０１１５】

これらの機構によってあるプロセスのコンテキスト情報を偽造して、プロセスの動作を変更するような攻撃や、コンテキスト情報を解析する攻撃をきわめて困難にして、安全性を高めることができる。

【０１１６】

同時に、コンテキスト情報の暗号化機構は、データ暗号化と共通のものを使うことが可能になり、安価なプロセッサの提供が可能となる。

【０１１７】

説明が前後したが、暗号化プロセスの実行を再開するとき、ｒｃｖｒｒｅｇ（ｒｅｃｏｖｅｒ　ｒｅｇｉｓｔｅｒ）命令を発行する。

ｒｃｖｒｒｅｇ　Ｒａ，　Ｒｂ

この命令は２つのレジスタオペランドをとる。Ｒａには保存したコンテキストのアドレスを格納する。Ｒｂにはプロセス識別子を指定する。

【０１１８】

この命令を実行したプロセッサコアは、コンテキストのアドレスに、プロセス識別子のコンテキスト鍵を示すタグをつけて、データキャッシュにメモリ読み出

し要求を発行する。もしデータキャッシュに保存したコンテキスト情報が残っていれば、それがそのまま読み出されてレジスタに復帰され、そのままプロセスの実行が再開される。

【０１１９】

もしコンテキスト情報がフラッシュされていれば、復号化機能が、プロセスのコンテキスト鍵を鍵管理機能に要求し、そこから読み出されたコンテキスト鍵が復号化に使われ、データキャッシュにコンテキスト情報が平文で読み込まれ、レジスタ情報が復帰されてプロセスの実行が再開される。

【０１２０】

他のプロセスがデータキャッシュ上に平文で存在するコンテキスト情報を読みだそうとしても、タグが一致しないため、読み出すことはできず、コンテキスト情報は安全である。

【０１２１】

（第３の実施形態）

本実施形態では、第２の実施形態と相違する点を中心にして説明する。

【０１２２】

第３の実施形態では、共有メモリの高速処理を実現する実施形態を説明する。プロセス間の通信の実現には、多くの場合何らかの共有メモリが用いられる。第２の実施形態の機構により、２つのプロセスの間での安全な鍵交換手法（寺本特許）を行い、データ暗号鍵を共有することにより、プロセッサの暗号機能を利用した効率的かつ安全なプロセス間通信が実現できる。

【０１２３】

しかしながら、第２の実施形態の機構では、プロセッサ内に復号化されたキャッシュラインを、外部メモリへのフラッシュ無しに複数のプロセス間で共有することはできない。例えば、プロセスＡがレジスタＤ１に格納された鍵Ｋａでデータを復号化してデータキャッシュに読み込んだとする。このキャッシュラインには暗号化属性（＃ａ，Ｄ１）が付与される。別のプロセスＢがレジスタＤ１に鍵Ｋａを保持していて、そのキャッシュラインを参照すると、プロセスのメモリ参照タグの暗号化属性（＃ｂ，Ｄ１）はキャッシュラインの暗号化属性（＃ａ，Ｄ

１）と一致しないので、そのキャッシュラインはフラッシュされ、一度タグ（＃ａ，Ｄ１）に対応する暗号鍵Ｋａで暗号化されてメインメモリに書き出され、再度タグ（＃ｂ，Ｄ１）に対応する暗号鍵Ｋａで復号化されてキャッシュに読み込まれる。

【０１２４】

ここでは不要なメインメモリへのアクセスが２度起きている。

【０１２５】

このオーバヘッドを削減し、効率のよいメモリ参照を行う一実施形態を以下に説明する。

【０１２６】

図８に、本実施形態に係るマイクロプロセッサの全体構成例を示す。第２の実施形態との違いは、プロセッサコア２０１とキャッシュメモリ３０１，４０１との間に、鍵インデックス変換機能８０１が追加されたことである。

【０１２７】

図９に、鍵インデックス変換機能８０１と鍵管理部７０１のより詳細な構成例を示す。鍵インデックス変換機能８０１は、内部に間接鍵値参照テーブル８０２を持つ。鍵管理部７０１は、内部に実効鍵の値を保持する鍵値テーブル８０４を持つ。間接鍵値参照テーブルは、プロセス識別子と鍵タイプ識別子によって参照され、鍵値テーブル８０４のエントリへのインデックスを保持する。図９においては、鍵値テーブルのエントリ＃Ｎへの参照を、間接鍵値テーブルの（＃１，Ｄ２）でインデックスされるエントリが保持している。

【０１２８】

鍵値テーブルの管理は、実効鍵値テーブル管理機能８０５が行う。実効鍵値テーブル管理機能８０５は、テーブル内で鍵の値が一意になるよう、登録、削除処理を行う。また、鍵値テーブルには、エントリが有効な鍵値を保持しているかどうかを判定するため、鍵値への参照数を保持するフィールドが設けられている。

【０１２９】

図１０に鍵登録時の動作例を示し、図１１に鍵登録処理の手順例を示す。

【０１３０】

まず、プロセッサコア２０１から状態管理機能７０２へ、プロセス識別子と鍵タイプ、鍵値情報が提示される。鍵値は直接値が提示されるとは限らず、プログラム鍵の場合は配布鍵が格納されたアドレスが提示され、コンテキスト鍵の場合は鍵値は鍵管理部内で生成される。

【０１３１】

鍵値Ｋａは実効鍵値テーブル管理機能８０５に提示される。実効鍵値テーブル管理機能８０５は、鍵値をテーブルで検索し、一致するエントリが存在しない場合には新たに値を登録し、参照カウンタの値を０から１に増やす。ここで鍵値Ｋａはエントリ＃Ｎに登録されているので、その参照カウンタの値を１から２に増やす。ここでは、暗号鍵の共有にはテーブル上の鍵値の一意性が必要なことに留意しなければならない。一意性を保証するには、テーブルの値を検索する必要が生じるが、その実装には連想メモリ（ＣＡＭ）を使う方法、平行ツリー（ＡＶＬｔｒｅｅ）を使うなどの検索の高速化手法が可能である。もし鍵を新たに登録することができなかった場合、状態管理機能７０２へ通知する。

【０１３２】

登録に成功すると、実効鍵値テーブル管理機能８０５は、鍵値が格納されたエントリへのインデックス＃Ｎを間接鍵値参照テーブル管理機能８０３へ提示する。間接鍵値参照テーブル管理機能８０３は、状態管理機能７０２からプロセス識別子と鍵タイプ（＃ｎ，Ｄ１）を提示されており、これでインデックスされたエントリに、鍵値のインデックス＃Ｎを格納する。格納の完了は状態管理機能７０２に通知される。

【０１３３】

（鍵値の参照）

プロセス＃１がデータ鍵Ｄ２によって参照するとき、プロセッサコアからは、プロセス識別子、鍵タイプ（＃１，Ｄ２）が鍵インデックス変換機能８０１に提示される。鍵インデックス変換機能８０１は、間接鍵値参照テーブルを参照して、このプロセス識別子、鍵タイプを鍵値インデックスに対応するタグ＃Ｎに変換して、キャッシュタグとしてデータキャッシュに提示する。データキャッシュは、キャッシュミスヒットの場合、アドレスに対応する外部メモリの内容を読み込

み、タグ＃Ｎに対応する鍵で復号化されたデータをキャッシュラインに読み込み、タグ＃Ｎを付加する。

【０１３４】

このキャッシュラインを別のプロセス＃ｎがデータ鍵Ｄ１によって参照するとき、鍵インデックス変換機能８０１が出力するタグは同一の＃Ｎとなり、復号化されたキャッシュラインが外部メモリを参照することなく利用できる。異なるプロセスが同一のタグを利用するには、実効鍵値テーブルに登録された鍵の値が同一である必要があるため、鍵の値を知らないプロセスが不正に復号化されたデータを読み出すことはない。

【０１３５】

（鍵エントリ削除処理）

図１２に鍵エントリ削除時の動作例を示し、図１３に鍵エントリ削処理の手順例を示す。

【０１３６】

プロセッサコア２０１は、削除する鍵に対応するプロセス識別子、鍵タイプ（＃ｎ，Ｄ１）を状態管理機能７０２に提示する。状態管理機能７０２は、間接鍵値参照テーブル管理機能８０３にプロセス識別子、鍵タイプ（＃ｎ，Ｄ１）を提示し、間接鍵値参照テーブル管理機能８０３は、テーブルから鍵値テーブルインデックス＃Ｎを取得し、同時に値を＃０に書き換える。間接鍵値参照テーブル管理機能８０３は、鍵値テーブルインデックス＃Ｎを実効鍵値テーブル管理機能８０５に提示し、削除を要求する。実効鍵値テーブル管理機能８０５は、＃Ｎの参照カウンタの値を２から１に減らして削除が完了すると、状態管理機能７０２に通知する。

【０１３７】

（第４の実施形態）

本実施形態では、第２の実施形態と相違する点を中心にして説明する。

【０１３８】

第４の実施形態では、実効鍵の外部メモリへの保存による、プロセスの永続化制御機能と、安全なフィードバック情報の保存機能を説明する。

【０１３９】

第２の実施形態のマイクロプロセッサでは、コンテキスト情報を暗号化して外部メモリに保存するための鍵情報（コンテキスト鍵）がプロセッサ内部のテーブル７０９に保持されていた。ここで、コンテキスト鍵はコンテキスト保存の都度更新されるため、一度保存されたコンテキスト情報を再度利用することはできなくなり、攻撃者が、攻撃者がコンテキスト情報を別のプロセスや同じプロセスの別の時点で保存したものにすりかえる攻撃を防ぐことができる。

【０１４０】

しかしながら、コンテキスト鍵がプロセッサ内部に保持されていることには次の２種類の欠点もある。ひとつの問題は、プログラムの動作に不具合が発見されたとき、プログラムの開発者にとっても、プログラムがどのような状態で実行されていたかがわからない、すなわちプログラム開発者へのフィードバックが不可能なことである。

【０１４１】

もうひとつの問題は、コンテキスト鍵テーブルの数を越えたプロセスを同時に実行することや、プロセッサの電源断をはさんで同一のプロセスの実行を継続することができず、利便性が損なわれることである。

【０１４２】

特に後者は上記のプロセスのコンテキストすり替えを防ぐセキュリティ上の安全性とユーザの利便性のトレードオフとなる。

【０１４３】

この２つの問題を解決する実施形態を以下に示す。

【０１４４】

図１４に、本実施形態に係るマイクロプロセッサの全体構成例を示す。第２の実施形態の場合に対して、プロセス状態入出力機能９０１が付加されている。プロセス状態入出力機能９０１には、プロセス状態鍵暗号・復号化機能９０２とプロセス状態保存用秘密鍵９０３がある。プロセス状態鍵暗号・復号化機能９０２は共通鍵アルゴリズムの暗号化・復号化機能を行うものであり、プロセス状態保存用秘密鍵９０３はプロセッサ固有であるとする。

【０１４５】

第４の実施形態では、プログラムの配布鍵として図１５に示す形式の情報がプロセッサ公開鍵により暗号化されて配布される。フィードバック鍵と永続化フラグの２つの情報が付加されている。また、ハッシュフィールドがもうけられ、命令鍵、フィードバック鍵、永続化フラグに対するハッシュ値が合致しない場合鍵のテーブルへの登録は中止される。配布鍵はプロセッサの秘密鍵により復号化され、命令鍵テーブルに格納される。命令鍵テーブルの形式を図１６に示す。命令鍵テーブル１５０１には＃０～＃ｎのエントリがある。エントリには命令鍵格納フィールド１５０１－０－ｐ，フィードバック鍵１５０１－０－ｆ，永続化制御フラグ１５０１－０－ｅがある。

【０１４６】

（フィードバック情報）

フィードバック鍵は、プログラム開発者が任意に決める共通鍵である。プログラム（通常はＯＳ）があるプロセスのフィードバック情報を保存するときにはフィードバック情報保存命令を発行する。保存対象のプロセスはコンテキストが保存され、実行が中断されていることが前提とする。フィードバック情報保存命令はオペランドとしてプロセス識別子と保存先メモリアドレスをとる。フィードバック情報保存命令の発行により、プロセスの鍵情報（命令鍵、コンテキスト鍵、データ鍵）はフィードバック鍵によって暗号化され、フィードバック情報としてプロセッサ外部メモリの指定アドレスに保存される。

【０１４７】

フィードバック鍵を知るプログラム開発者はフィードバック情報を復号化して、中断時のプロセスの状態を知ることができるが、プロセッサにはフィードバック情報から鍵情報を復元する機構をそなえない。このため、フィードバック情報がプロセスにあやまった状態を与えるために悪用されるおそれはなく、フィードバック情報から命令鍵やコンテキスト鍵を特定することもフィードバック鍵を知らないユーザには不可能である。

【０１４８】

（永続化制御）

図１７に、永続化のためのプロセス鍵の保存手順の一例を示す。

【０１４９】

また、配布鍵の永続化フラグもプログラム開発者が決める情報である。プログラムの利便性を重視して永続化を許すとき、プログラム開発者は永続化フラグを１とした配布鍵を提供する。プログラムの安全性を重視して永続化を許さないとき、プログラム開発者は永続化フラグを０として配布鍵を提供する。

【０１５０】

あるプロセスの鍵情報を保存するときは、プロセス状態保存命令をプロセス識別子と保存先メモリアドレスをオペランドとして発行する。プロセス状態入出力機能は、命令鍵テーブルのプロセス識別子に対応するエントリを参照して、永続化フラグが１のときのみ、プロセス状態保存処理を行い、永続化フラグが０の場合は例外を発生させる。

【０１５１】

保存処理では、プロセスの鍵情報（命令鍵、コンテキスト鍵、データ鍵）はプロセス状態保存用秘密鍵によって暗号化され、指定された外部メモリアドレスに書き出される。

【０１５２】

保存したプロセスの鍵情報を復帰するときは、プロセス状態復帰命令をプロセス識別子と保存先メモリアドレスをオペランドとして発行する。プロセッサはプロセス状態保存用秘密鍵により、プロセス鍵を復号化して、指定したプロセス識別子に対応するテーブルに登録する。その後、対応するコンテキスト情報とプロセス識別子を指定してプロセス再開命令を発行することにより、プロセスの実行が再開される。

【０１５３】

保存されたプロセス状態は、プロセッサ固有のプロセス状態保存用秘密鍵によって暗号化されているので、他のプロセッサに対してプロセス状態を移動することはできず、プログラムの不正利用の可能性を低減することができる。

【０１５４】

永続化制御フラグは配布鍵に含まれ、暗号化とハッシュによって守られている

ため、永続化フラグを変更することは正しい命令鍵を知らない限り不可能である。そして、復号化された永続化制御フラグにもとづいて、プロセス状態入出力機能によるプロセス鍵情報の保存処理の可否を判断することにより、プロセスの永続性と、安全性という相反する要素を、（一般に）プログラムの著作権者であるプログラム開発者が制御可能とする効果が本発明により実現される。

【０１５５】

なお、以上の各機能は、ソフトウェアとして実現可である。

また、本実施形態は、コンピュータに所定の手段を実行させるための（あるいはコンピュータを所定の手段として機能させるための、あるいはコンピュータに所定の機能を実現させるための）プログラムとして実施することもでき、該プログラムを記録したコンピュータ読取り可能な記録媒体として実施することもできる。

【０１５６】

なお、この発明の実施の形態で例示した構成は一例であって、それ以外の構成を排除する趣旨のものではなく、例示した構成の一部を他のもので置き換えたり、例示した構成の一部を省いたり、例示した構成に別の機能あるいは要素を付加したり、それらを組み合わせたりすることなどによって得られる別の構成も可能である。また、例示した構成と論理的に等価な別の構成、例示した構成と論理的に等価な部分を含む別の構成、例示した構成の要部と論理的に等価な別の構成なども可能である。また、例示した構成と同一もしくは類似の目的を達成する別の構成、例示した構成と同一もしくは類似の効果を奏する別の構成なども可能である。

また、この発明の実施の形態で例示した各種構成部分についての各種バリエーションは、適宜組み合わせて実施することが可能である。

また、この発明の実施の形態は、プロセッサとしての発明、プロセッサ内部の構成部分についての発明、またはそれらに対応する方法の発明等、種々の観点、段階、概念またはカテゴリに係る発明を包含・内在するものである。

従って、この発明の実施の形態に開示した内容からは、例示した構成に限定されることなく発明を抽出することができるものである。

【０１５７】

本発明は、上述した実施の形態に限定されるものではなく、その技術的範囲において種々変形して実施することができる。

【０１５８】

【発明の効果】

本発明によれば、キャッシュメモリ管理手順において、初期登録における公開鍵処理時間による応答不感時間を短縮して、リアルタイム処理性能を向上させることのできるマイクロプロセッサを提供することができる。

【０１５９】

また、本発明によれば、キャッシュメモリのアクセス制御に使われるタグの安全な管理を可能にしたマイクロプロセッサを提供することができる。

【０１６０】

また、本発明によれば、連鎖によってプログラムの暗号鍵を推定されにくくすると同時に効率的な命令実行を実現するプログラム復号化手段を持つマイクロプロセッサを提供することができる。

【０１６１】

また、本発明によれば、複数のプロセスが同一の暗号鍵によって暗号化されているデータを共有をするときの処理効率を向上させることのできるマイクロプロセッサを提供することができる。

【０１６２】

また、本発明によれば、安全なプログラム利用の利便性と秘密の安全性という、相反する要素を、プログラム提供者の意志にしたがって実現するコンテキスト保存手段を持つマイクロプロセッサを提供することができる。

【図面の簡単な説明】

【図１】

本発明の第１の実施形態に係るマイクロプロセッサの全体構成例を示す図

【図２】

プログラムのメモリ上配置について説明するための図

【図３】

同実施形態の鍵管理部の構成例を示す図

【図４】

同実施形態の鍵登録処理の一例を示すフローチャート

【図５】

同施形態の命令復号化処理部の構成例を示す図

【図６】

本発明の第２の実施形態に係るマイクロプロセッサの基本構成例を示す図

【図７】

同実施形態の鍵管理部の構成例を示す図

【図８】

本発明の第３の実施形態に係るマイクロプロセッサの基本構成例を示す図

【図９】

同実施形態の鍵管理部の構成例を示す図

【図１０】

同実施形態の鍵登録処理について説明するための図

【図１１】

同実施形態の鍵登録処理の一例を示すフローチャート

【図１２】

同実施形態の鍵削除処理について説明するための図

【図１３】

同実施形態の鍵削除処理の一例を示すフローチャート

【図１４】

本発明の第４の実施形態に係るマイクロプロセッサの鍵管理部の構成例を示す図

【図１５】

同施形態の配布鍵形式の一例を示す図

【図１６】

同実施形態の命令鍵テーブル形式の一例を示す図

【図１７】

同実施形態のプロセス鍵保存手順の一例を示すフローチャート

【符号の説明】

１０１…プロセッサ

１０２…内部データバス

１０３…外部バスインタフェース

２０１…プロセッサコア

２０２…実行鍵識別子格納レジスタ

２０３…制御レジスタ

２０４…レジスタ・制御レジスタ

２０５…プロセス識別子格納レジスタ

２０６…データ暗号化制御レジスタ

３０１…命令キャッシュ

４０１…データキャッシュ

５０１…復号化処理部

５０２…アドレス処理部

５０３…復号化機能

５０４…バッファ

５０５…連鎖計算機能

６０１…データ暗号化・復号化処理部

７０１…鍵管理部

７０２…状態管理機能

７０２－１…プロセス状態管理テーブル

７０３…配布鍵読み出し機能

７０４…秘密鍵

７０５…実効鍵復号化機能

７０６…鍵テーブル

７０７…命令鍵管理機能

７０８…乱数発生機能

７０９…コンテキスト鍵テーブル

７１０…データ鍵テーブル

８０１…鍵インデックス変換機能

８０２…間接鍵値参照テーブル

８０３…間接鍵値参照テーブル管理機能

８０４…鍵値テーブル

８０５…実効鍵値テーブル管理機能

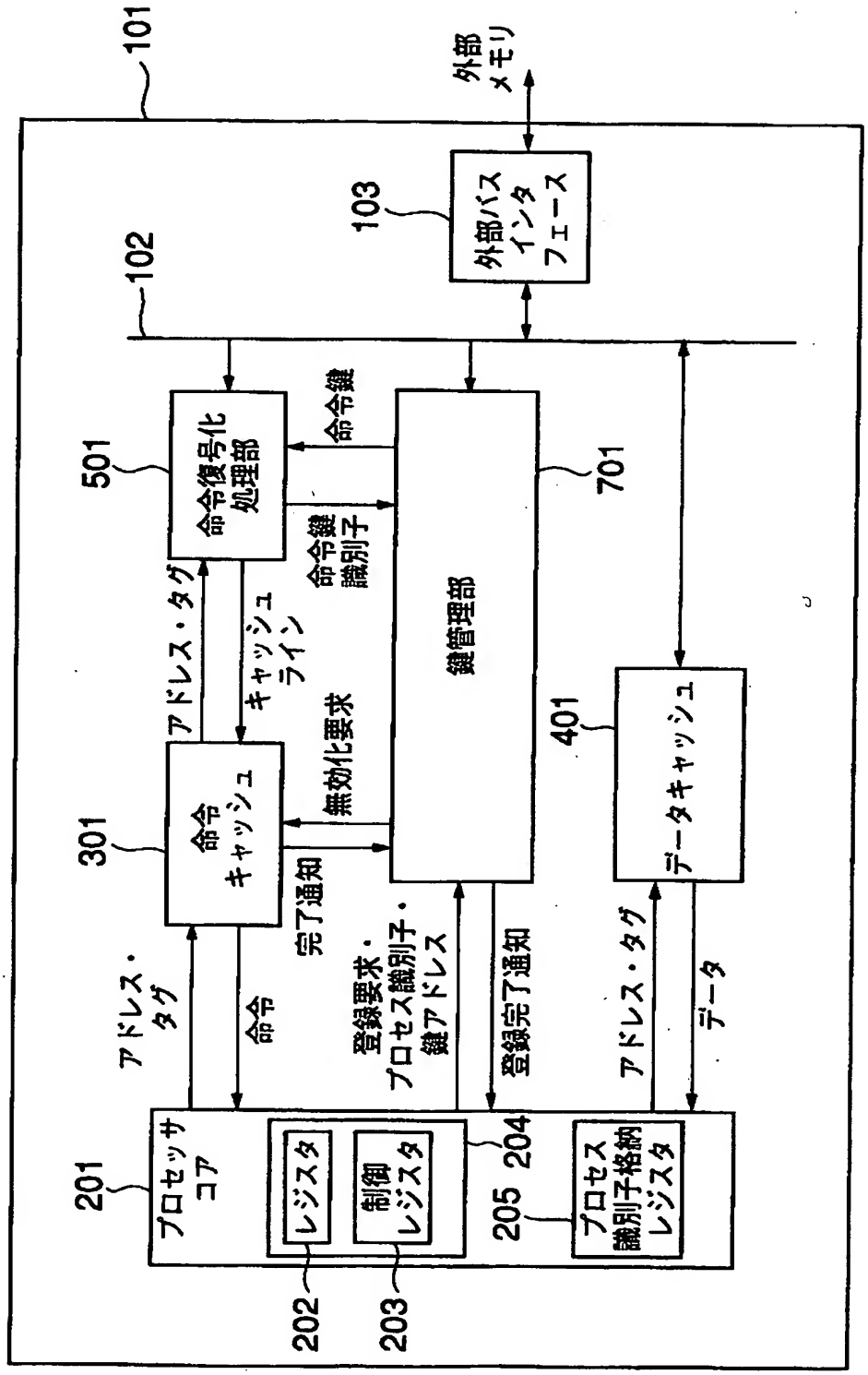
９０１…プロセス状態入出力機能

９０２…プロセス状態保存用秘密鍵

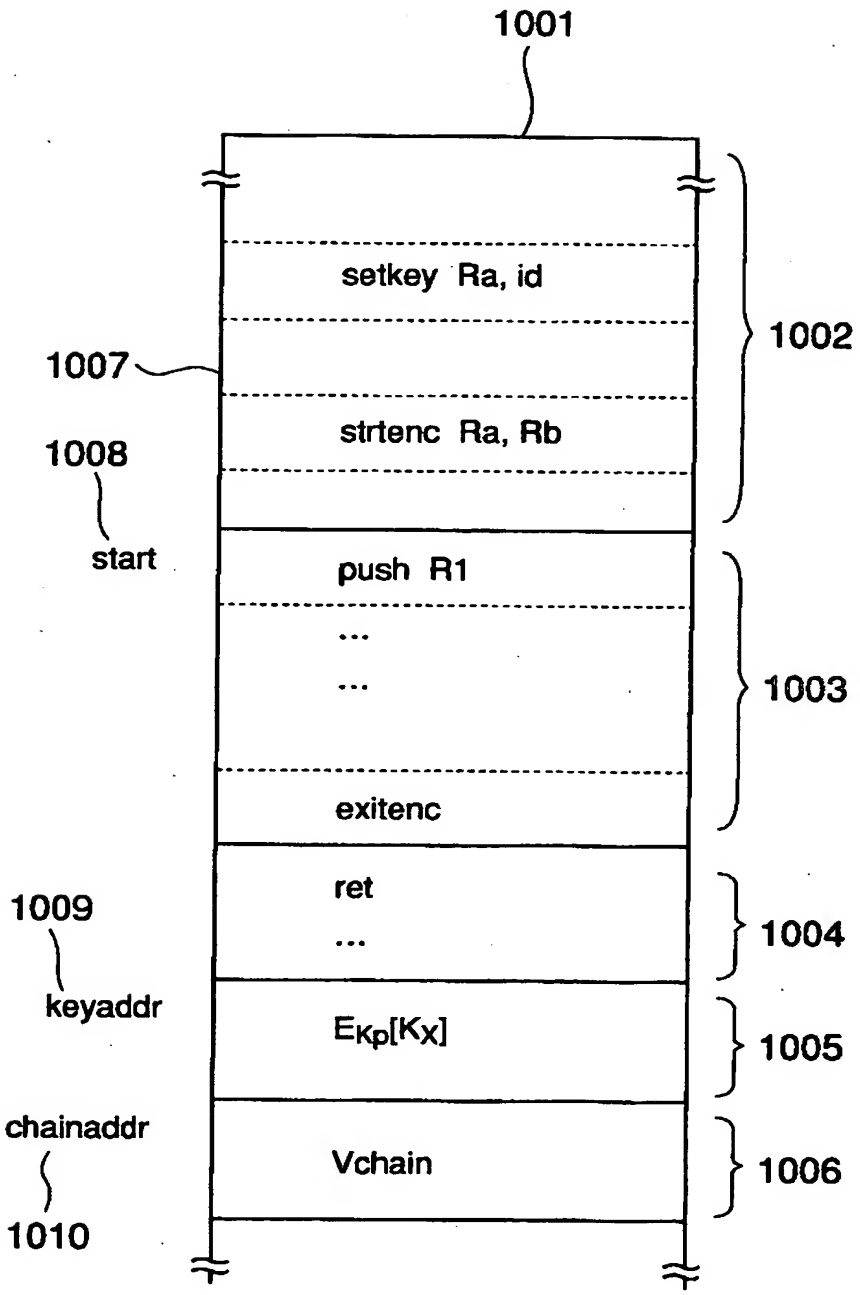
９０３…プロセス状態鍵暗号・復号化機能

【書類名】　　図面

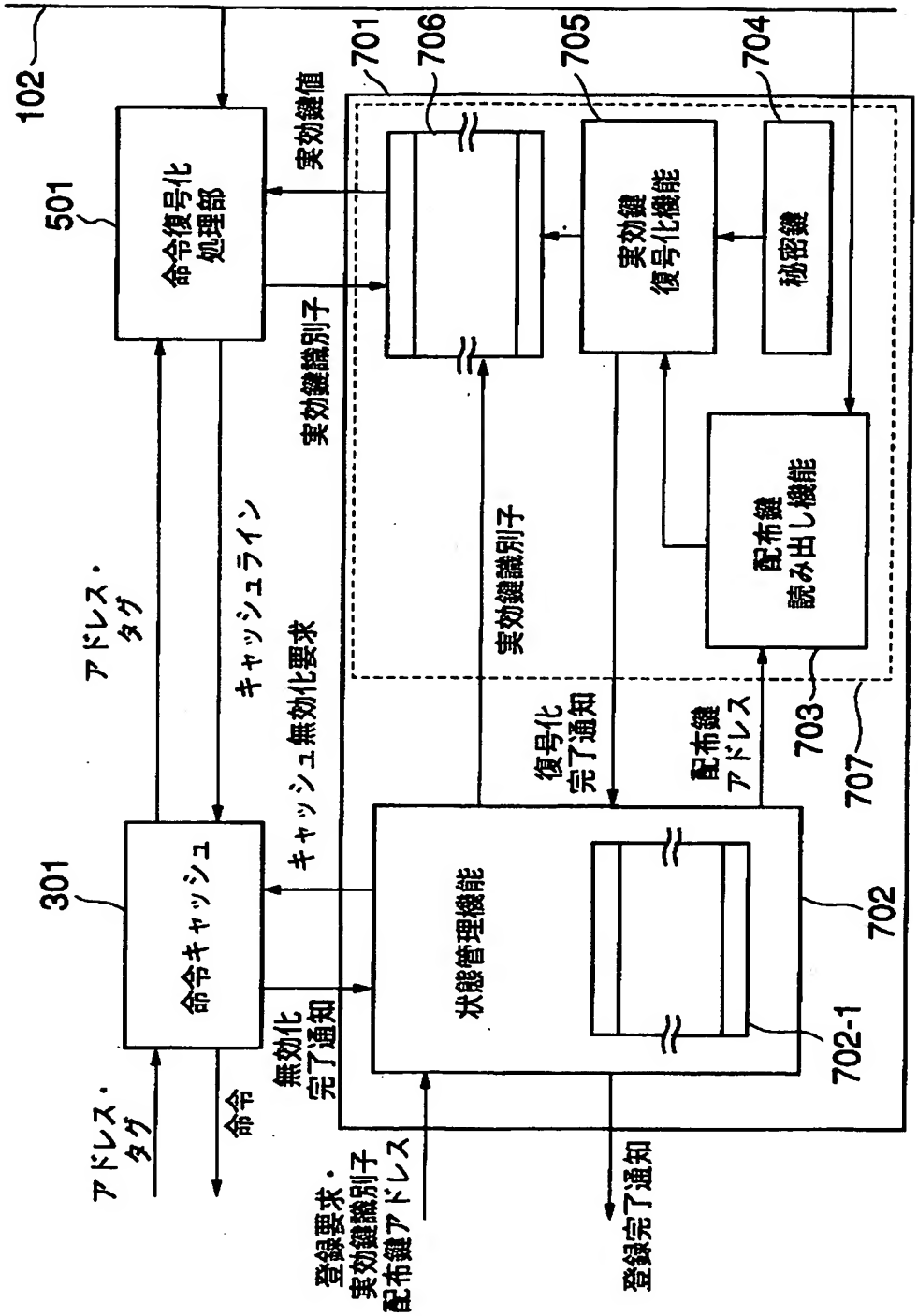
【図１】


101
外部メモリ
103
外部バスインタフェース
102
501
命令復号化処理部
命令鍵
命令鍵識別子
701
鍵管理部
アドレス・タグ
キャッシュライン
無効化要求
401
データキャッシュ
301
命令キャッシュ
完了通知
登録要求・プロセス識別子・鍵アドレス
登録完了通知
アドレス・タグ
データ
アドレス・タグ
命令
201
プロセッサコア
レジスタ
制御レジスタ
204
205
プロセス識別子格納レジスタ
202
203

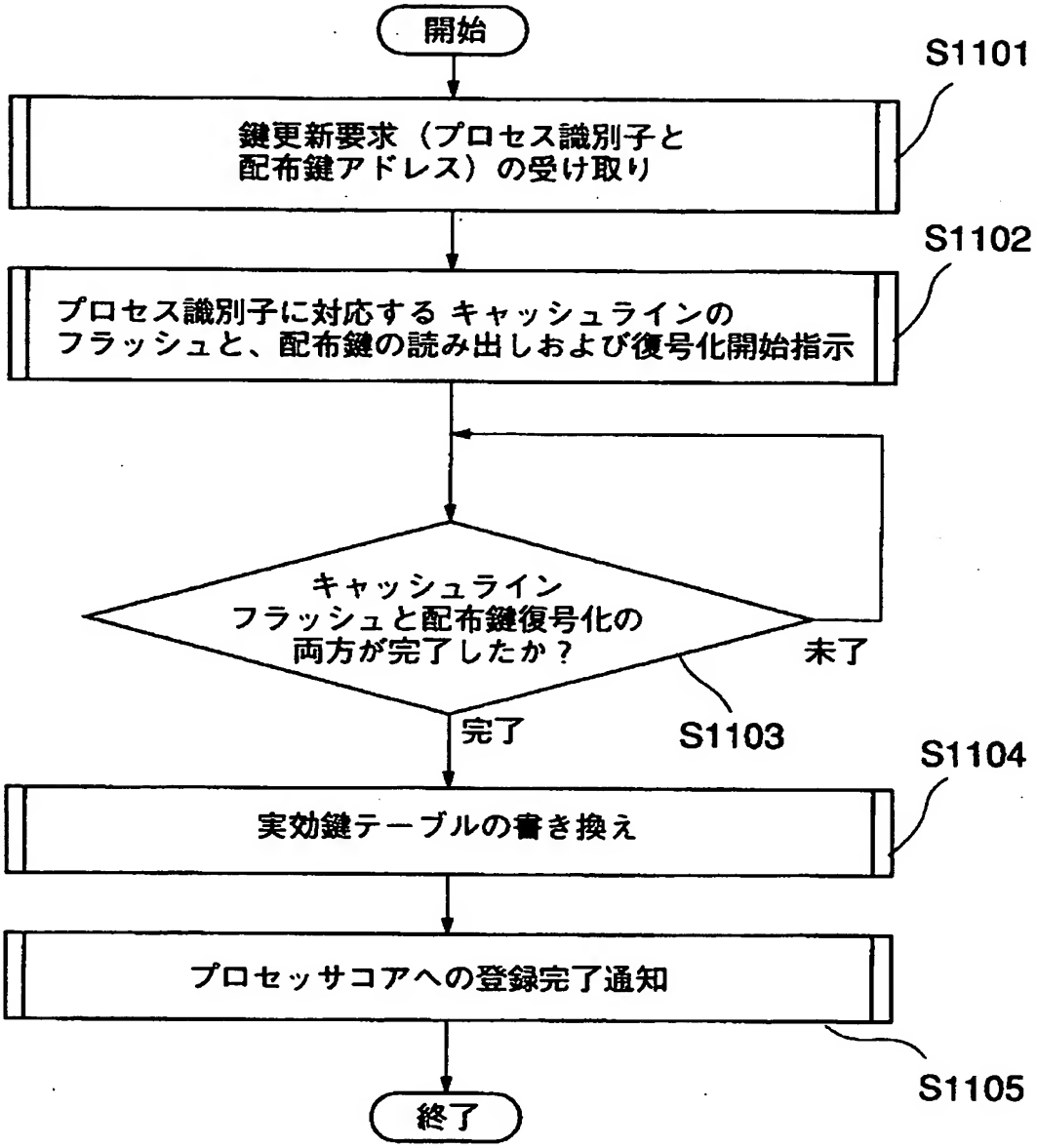
【図２】


1001
setkey Ra, id
1002
1007
strtenc Ra, Rb
1008
start
push R1
...
...
1003
exitenc
ret
...
1004
1009
keyaddr
EKp[KX]
1005
chainaddr
Vchain
1006
1010

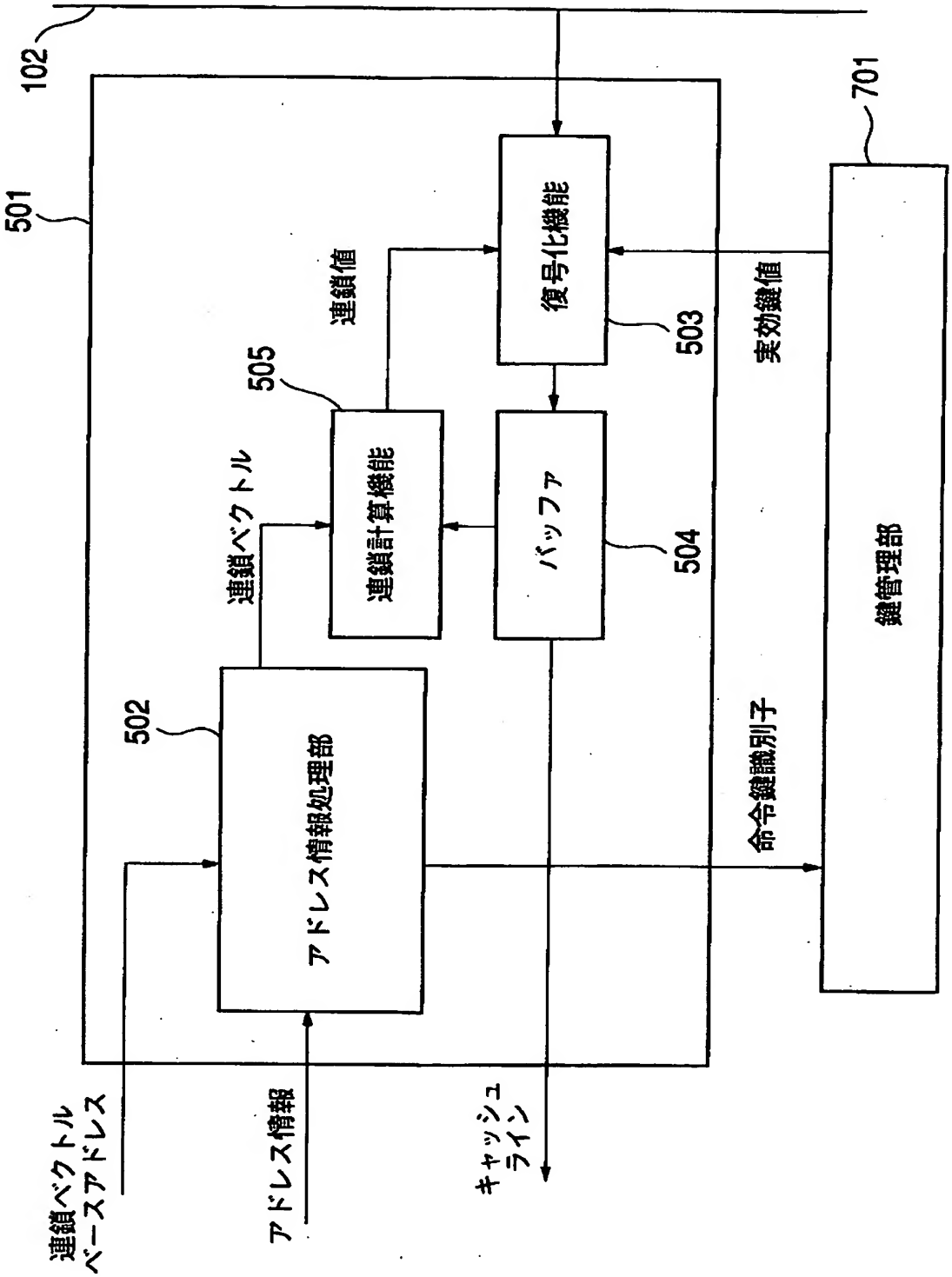
【図３】


102
701
706
705
704
501
命令復号化
処理部
実効鍵値
実効鍵
復号化機能
秘密鍵
実効鍵識別子
実効鍵識別子
配布鍵
読み出し機能
アドレス・
タグ
キャッシュライン
キャッシュ無効化要求
復号化
完了通知
配布鍵
アドレス
703
707
301
命令キャッシュ
状態管理機能
702
702-1
無効化
完了通知
アドレス・
タグ
命令
登録要求・
実効鍵識別子
配布鍵アドレス
登録完了通知

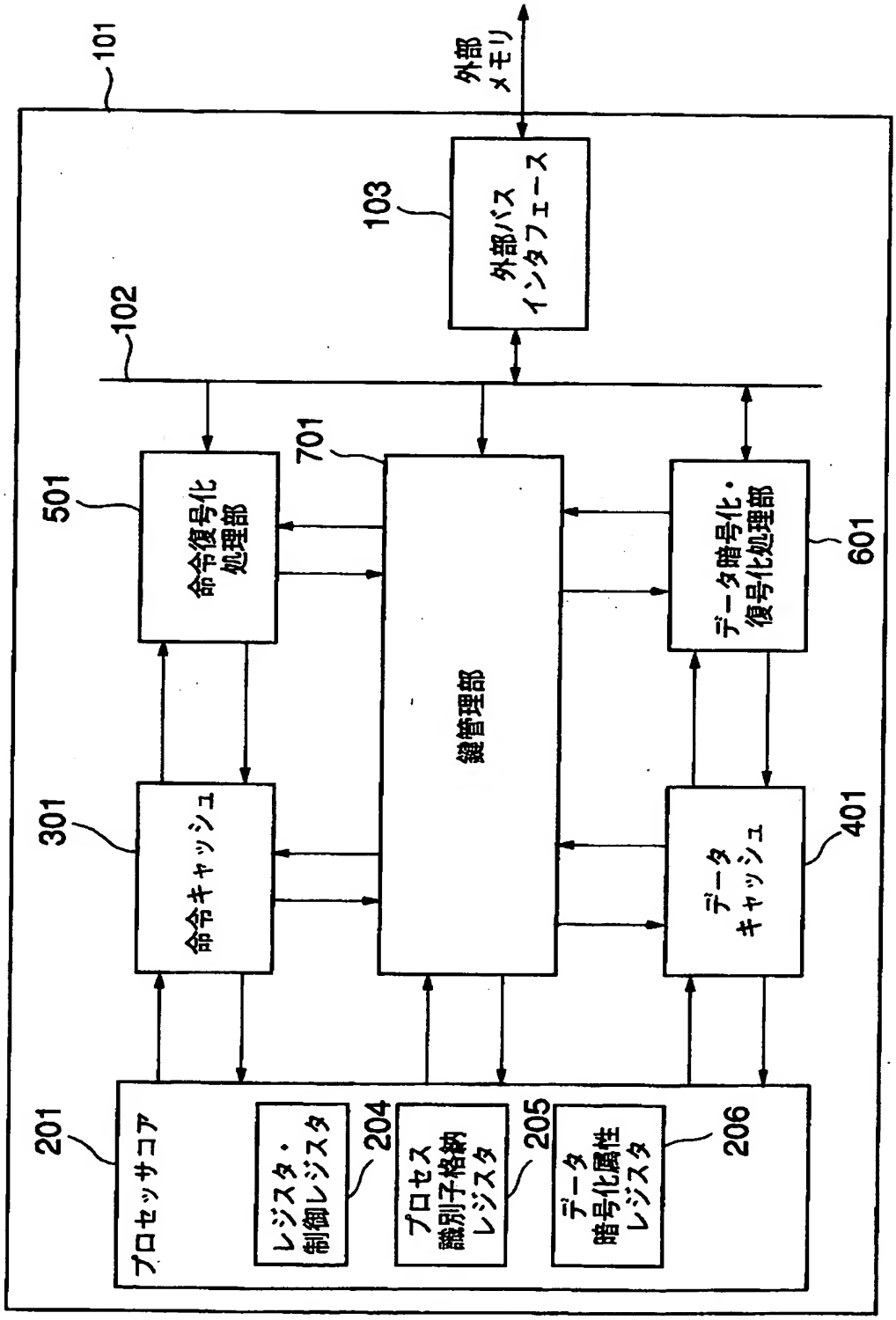
【図４】


開始
S1101
鍵更新要求（プロセス識別子と
配布鍵アドレス）の受け取り
S1102
プロセス識別子に対応する キャッシュラインの
フラッシュと、配布鍵の読み出しおよび復号化開始指示
キャッシュライン
フラッシュと配布鍵復号化の
両方が完了したか？
未了
完了
S1103
S1104
実効鍵テーブルの書き換え
プロセッサコアへの登録完了通知
S1105
終了

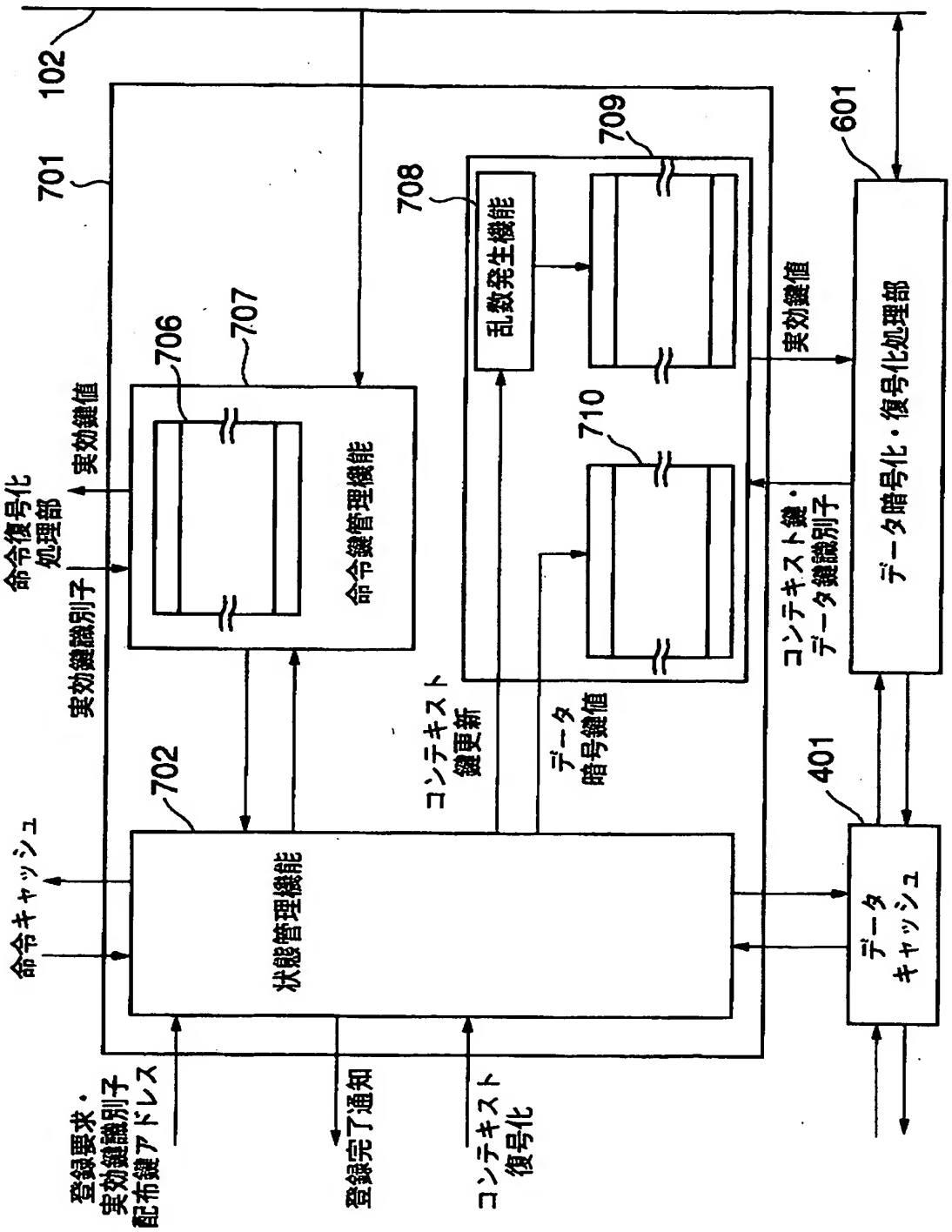
【図５】


102
501
701
復号化機能
503
実効鍵値
連鎖値
505
連鎖ベクトル
連鎖計算機能
バッファ
504
鍵管理部
502
アドレス情報処理部
命令鍵識別子
連鎖ベクトル
ベースアドレス
アドレス情報
キャッシュ
ライン

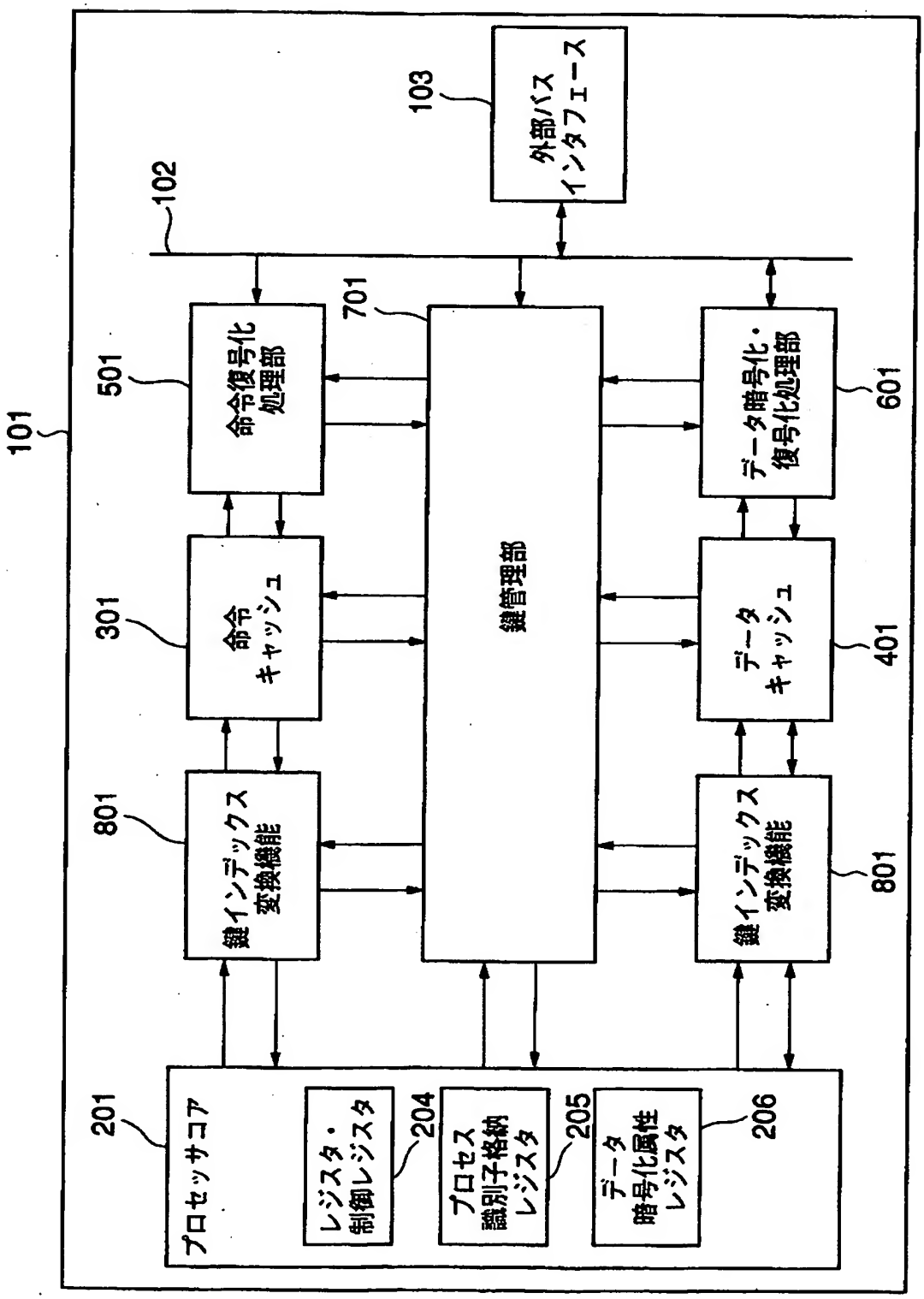
【図６】


101
外部メモリ
103
外部バスインタフェース
102
501
命令復号化処理部
701
鍵管理部
601
データ暗号化・復号化処理部
301
命令キャッシュ
401
データキャッシュ
201
プロセッサコア
204
レジスタ・制御レジスタ
205
プロセス識別子格納レジスタ
206
データ暗号化属性レジスタ

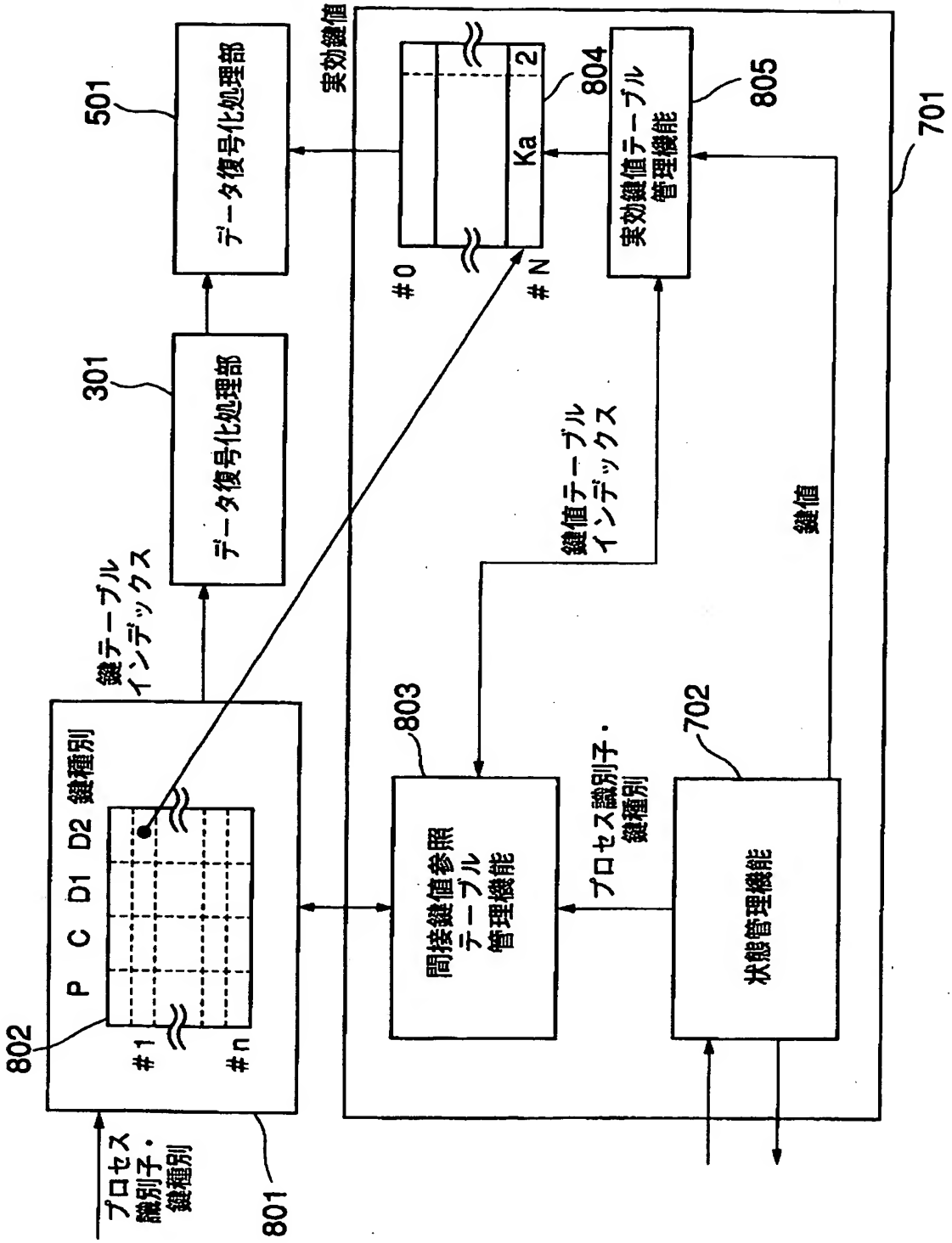
【図７】


102
701
702
706
707
708
709
710
601
401
命令復号化処理部
実効鍵識別子
実効鍵値
命令鍵管理機能
乱数発生機能
コンテキスト鍵更新
データ暗号鍵値
コンテキスト鍵・データ鍵識別子
データ暗号化・復号化処理部
命令キャッシュ
状態管理機能
データキャッシュ
登録要求・実効鍵識別子配布鍵アドレス
登録完了通知
コンテキスト復号化

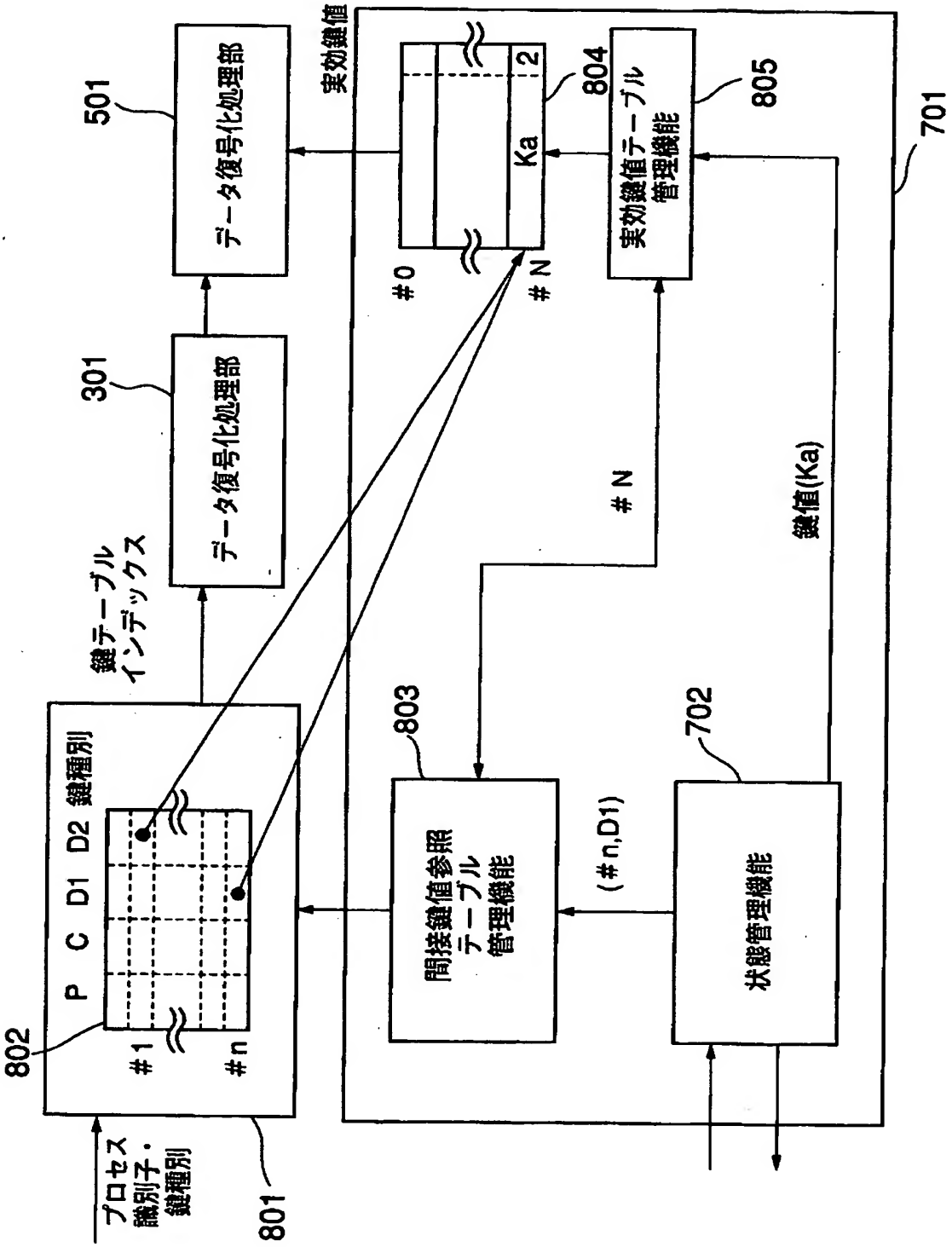
【図８】


101
102
103
外部バス
インタフェース
701
501
命令復号化
処理部
601
データ暗号化・
復号化処理部
301
命令
キャッシュ
鍵管理部
401
データ
キャッシュ
801
鍵インデックス
変換機能
801
鍵インデックス
変換機能
201
プロセッサコア
レジスタ・
制御レジスタ
204
プロセス
識別子格納
レジスタ
205
データ
暗号化属性
レジスタ
206

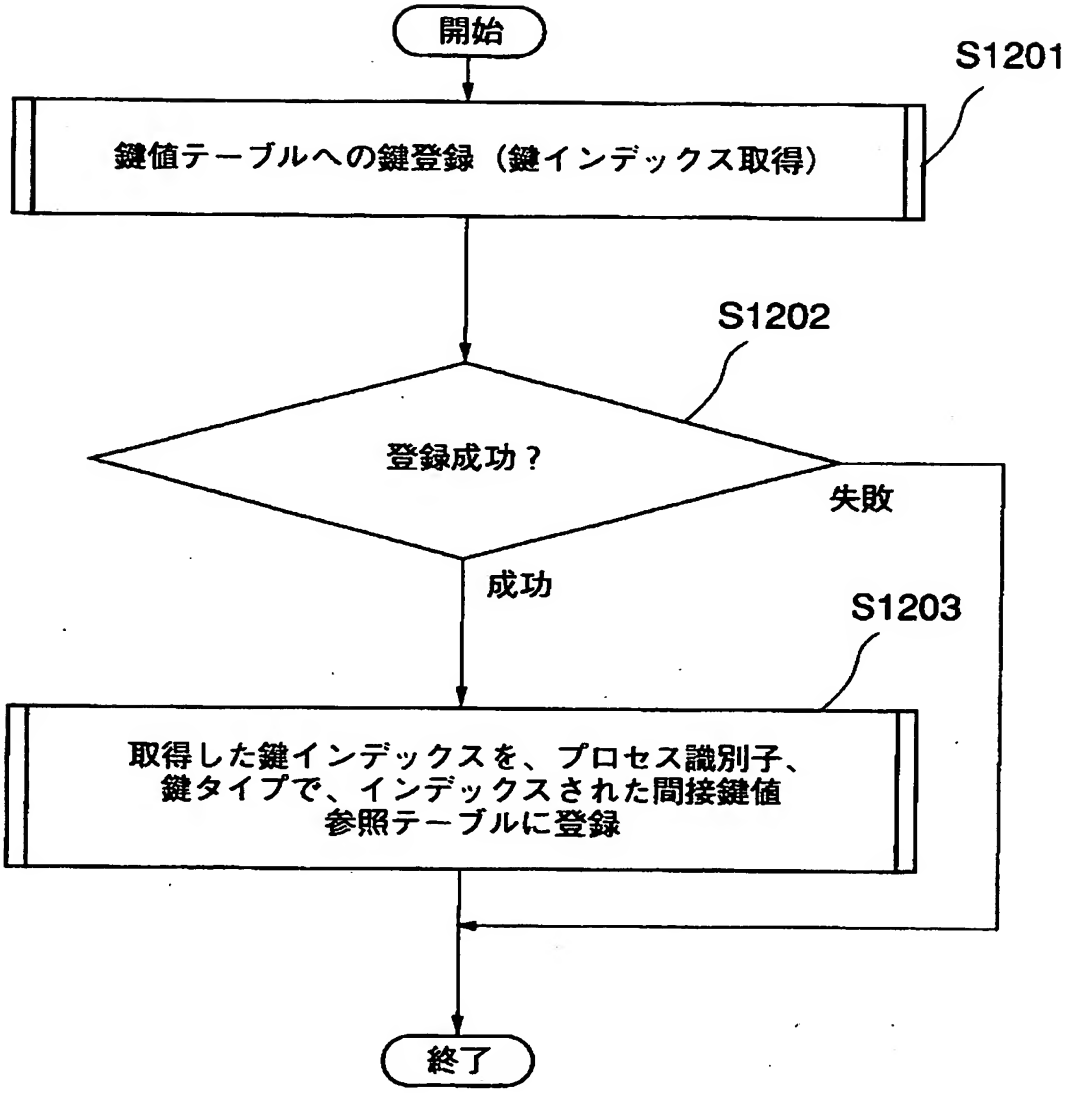
【図９】


501
データ復号化処理部
301
データ復号化処理部
実効鍵値
#0
Ka
2
#N
804
805
実効鍵値テーブル
管理機能
701
鍵テーブル
インデックス
鍵値テーブル
インデックス
鍵値
803
間接鍵値参照
テーブル
管理機能
プロセス識別子・
鍵種別
702
状態管理機能
802
P C D1 D2 鍵種別
#1
#n
プロセス
識別子・
鍵種別
801

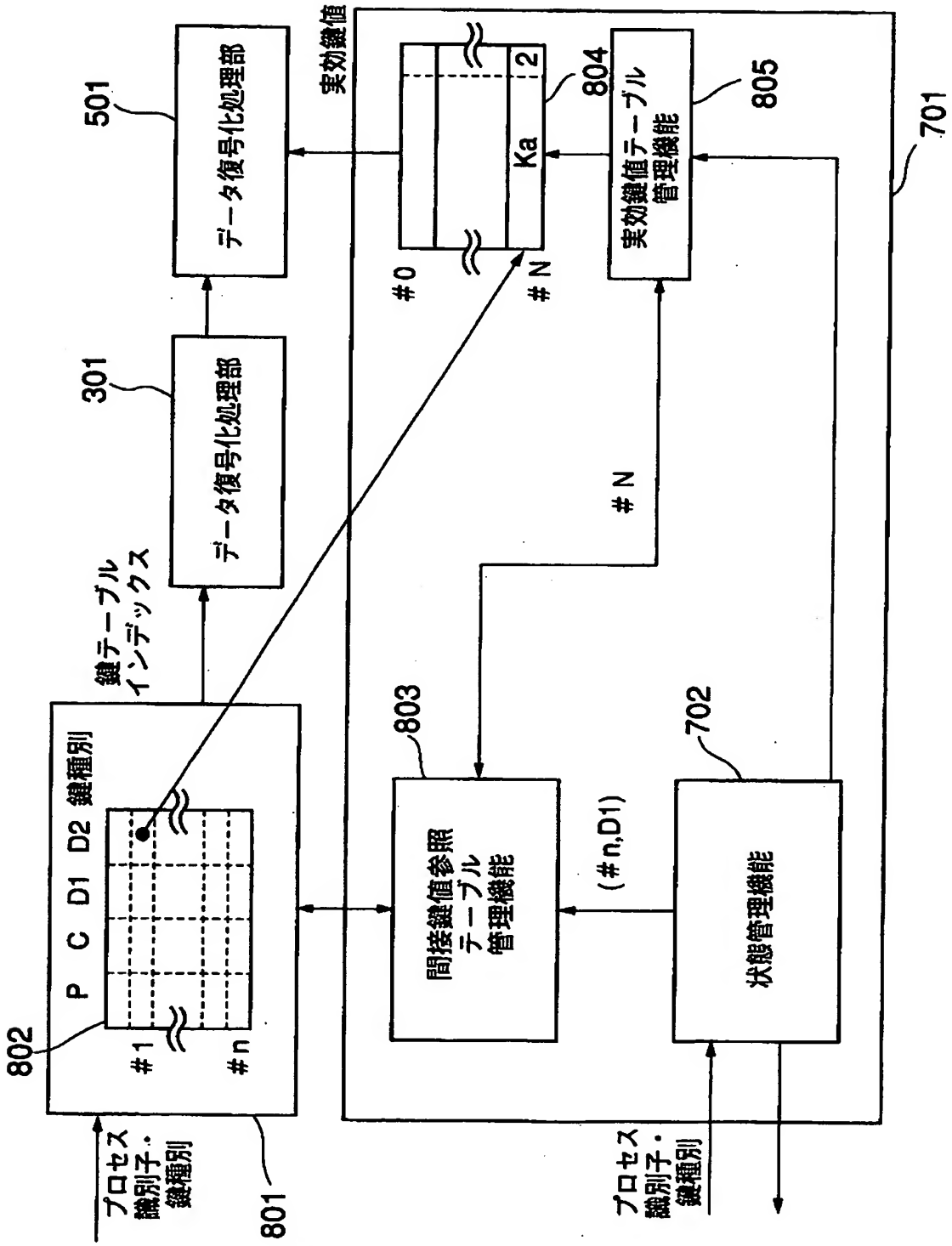
【図１０】


801
プロセス識別子・鍵種別
802
P C D1 D2 鍵種別
#1
#n
鍵テーブルインデックス
301
データ復号化処理部
501
データ復号化処理部
実効鍵値
#0
Ka
2
#N
804
実効鍵値テーブル管理機能
805
701
#N
鍵値(Ka)
803
間接鍵値参照テーブル管理機能
(#n,D1)
702
状態管理機能

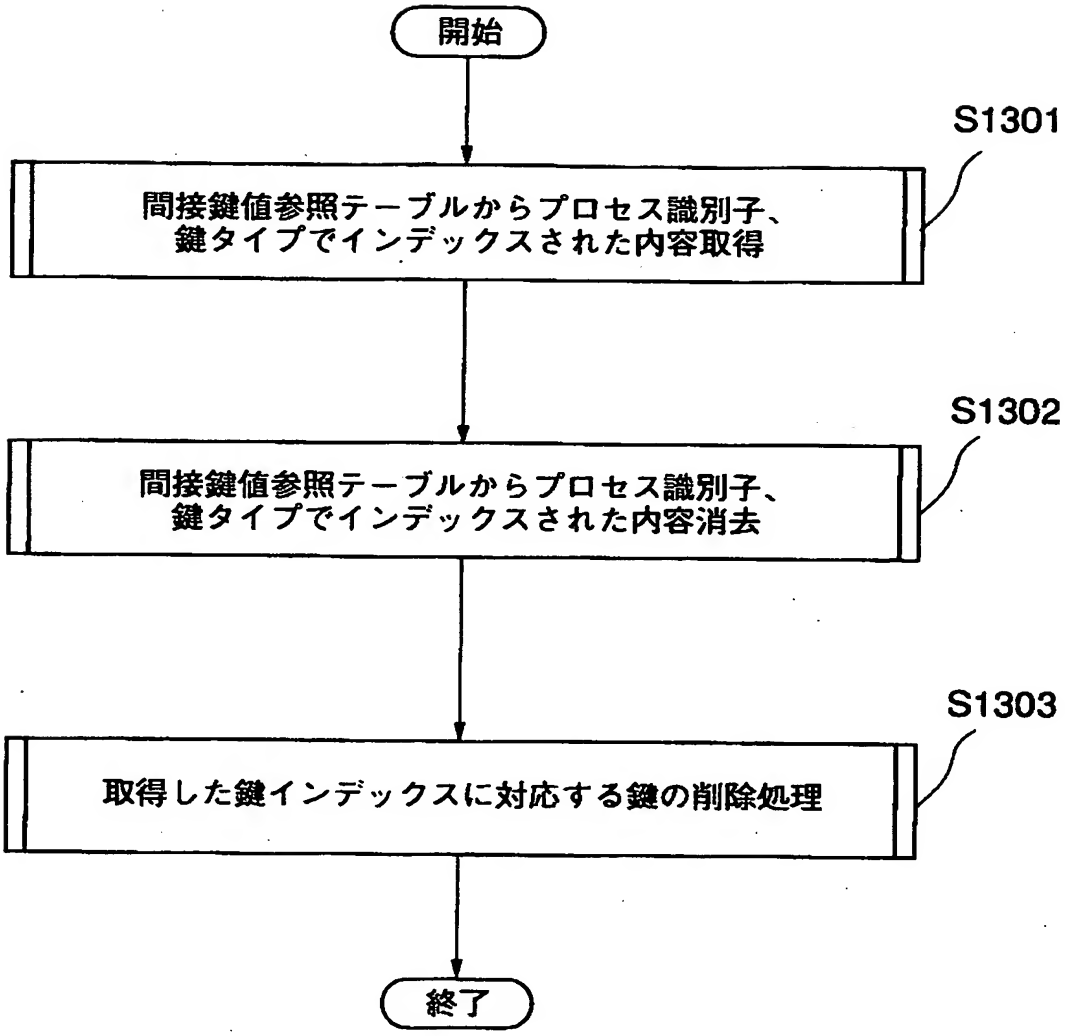
【図１１】


開始
S1201
鍵値テーブルへの鍵登録（鍵インデックス取得）
S1202
登録成功？
失敗
成功
S1203
取得した鍵インデックスを、プロセス識別子、
鍵タイプで、インデックスされた間接鍵値
参照テーブルに登録
終了

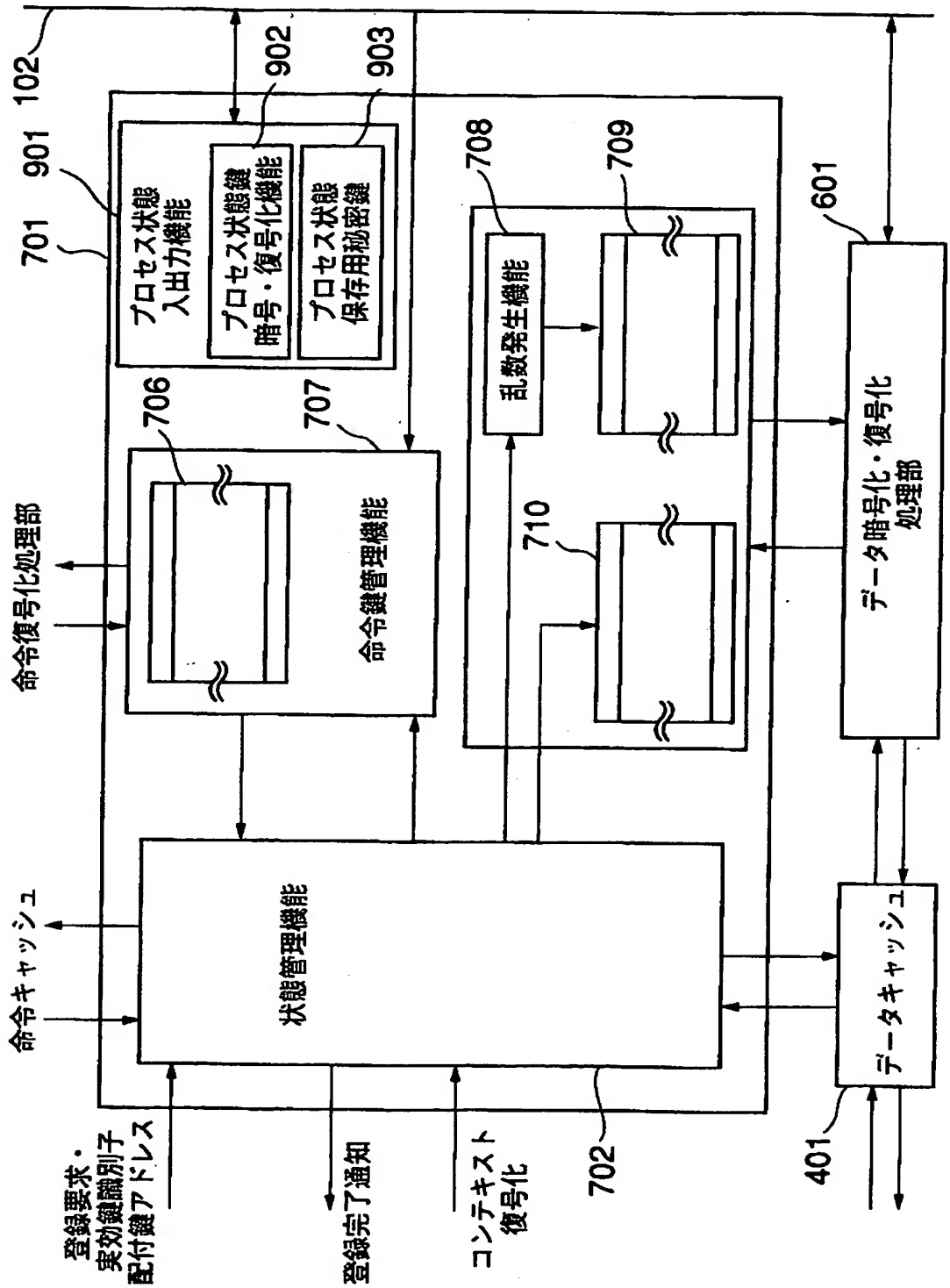
【図１２】


501
データ復号化処理部
301
データ復号化処理部
実効鍵値
#0
Ka
2
#N
804
実効鍵値テーブル
管理機能
805
701
#N
鍵テーブル
インデックス
803
間接鍵値参照
テーブル
管理機能
(#n,D1)
702
状態管理機能
P C D1 D2 鍵種別
802
#1
#n
プロセス
識別子・
鍵種別
801
プロセス
識別子・
鍵種別

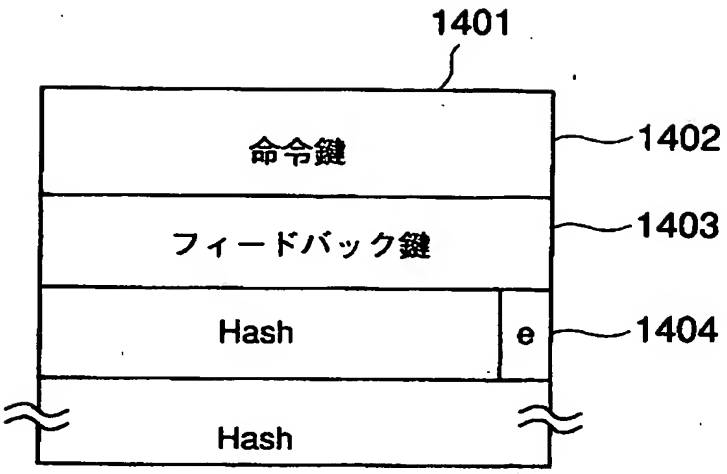
【図１３】


開始
S1301
間接鍵値参照テーブルからプロセス識別子、
鍵タイプでインデックスされた内容取得
S1302
間接鍵値参照テーブルからプロセス識別子、
鍵タイプでインデックスされた内容消去
S1303
取得した鍵インデックスに対応する鍵の削除処理
終了

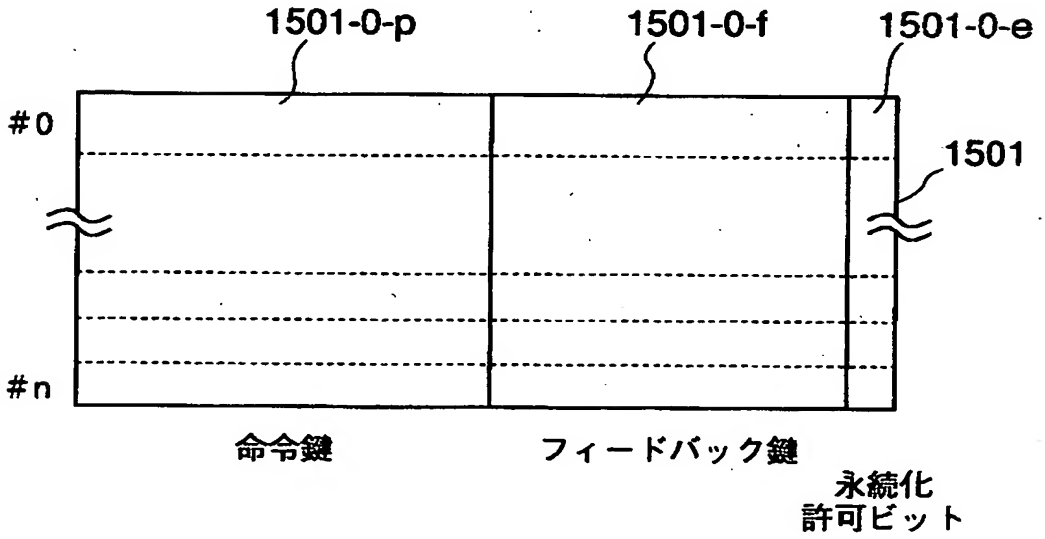
【図１４】


102
901
902
903
701
プロセス状態
入出力機能
プロセス状態鍵
暗号・復号化機能
プロセス状態
保存用秘密鍵
708
709
601
乱数発生機能
706
707
命令鍵管理機能
710
データ暗号化・復号化
処理部
命令復号化処理部
状態管理機能
命令キャッシュ
データキャッシュ
登録要求・
実効鍵識別子
配付鍵アドレス
登録完了通知
コンテキスト
復号化
702
401

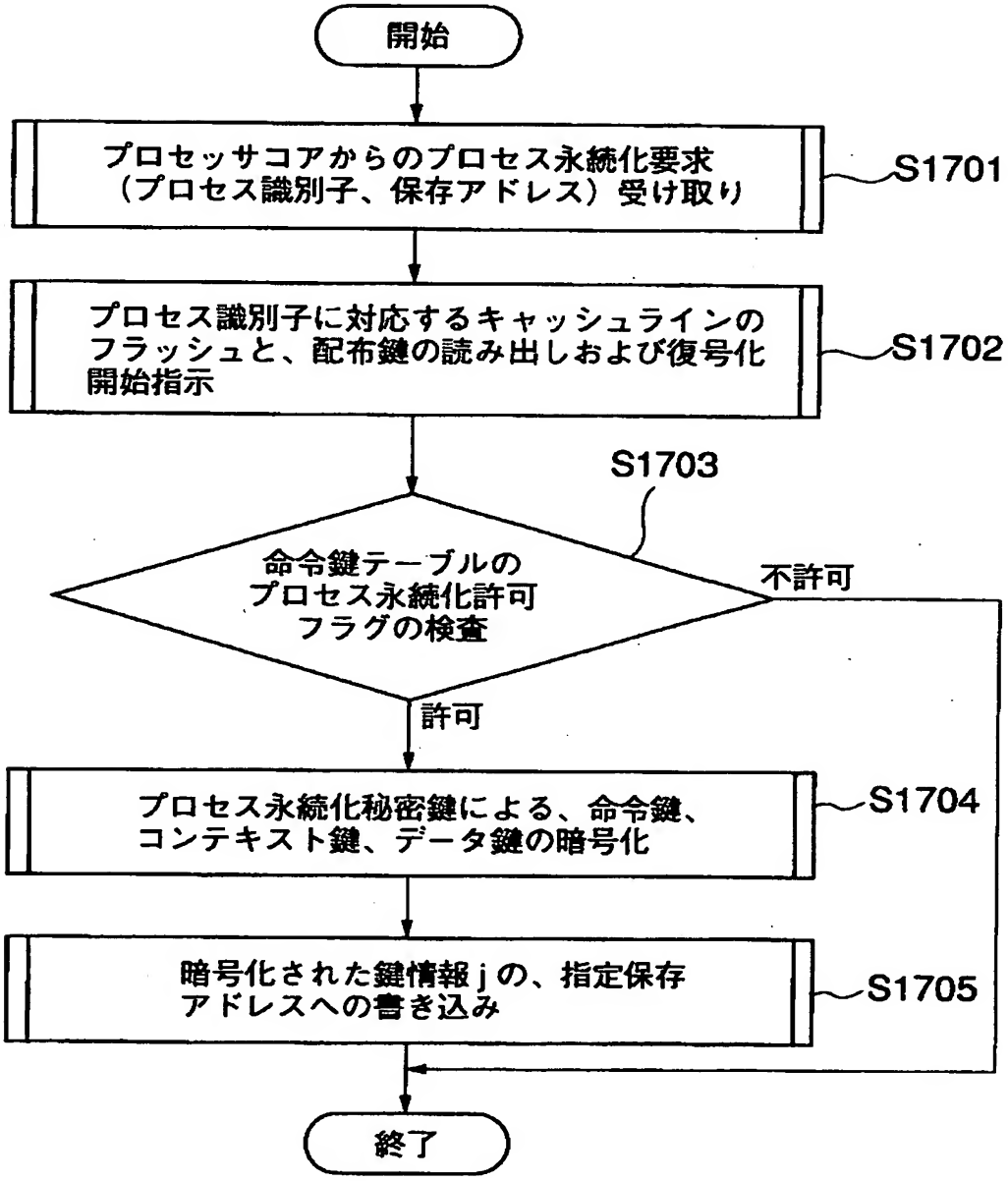
【図１５】


1401
命令鍵
1402
フィードバック鍵
1403
Hash
e
1404
Hash

【図１６】


1501-0-p
1501-0-f
1501-0-e
#0
1501
#n
命令鍵
フィードバック鍵
永続化
許可ビット

【図１７】


開始
プロセッサコアからのプロセス永続化要求
（プロセス識別子、保存アドレス）受け取り
S1701
プロセス識別子に対応するキャッシュラインの
フラッシュと、配布鍵の読み出しおよび復号化
開始指示
S1702
S1703
命令鍵テーブルの
プロセス永続化許可
フラグの検査
不許可
許可
プロセス永続化秘密鍵による、命令鍵、
コンテキスト鍵、データ鍵の暗号化
S1704
暗号化された鍵情報ｊの、指定保存
アドレスへの書き込み
S1705
終了

【書類名】　　要約書

【要約】

【課題】　リアルタイム処理性能を向上したプロセッサを提供すること。

【解決手段】　プロセッサ固有の秘密鍵に対する公開鍵で暗号化した命令鍵を外部メモリから読み出す機能、読み出した該鍵の秘密鍵での復号機能、プログラム識別子対応に命令鍵を保持するテーブルへの鍵登録機能、登録完了の命令実行部への通知機能、実行中プログラムの識別子の保持機能と、外部メモリの所定のアドレスの内容を第５機能にて指示されるテーブルに格納された鍵で復号する機能、この機能で復号した命令と外部メモリからの命令の実行機能、復号機能に命令鍵を復号させ該命令にて指定されたプログラム識別子と命令鍵を対応させてテーブルに登録しその完了を命令実行部に通知する処理と、保持機能に該命令において指定されるプログラム識別子を設定し該命令にて指定される外部メモリのアドレスにプログラム実行制御を移す処理を実行する機能を持つ。

【選択図】　　図１

出願人履歴情報

識別番号　〔０００００３０７８〕

１．変更年月日　１９９０年　８月２２日
［変更理由］　新規登録
住　所　神奈川県川崎市幸区堀川町７２番地
氏　名　株式会社東芝

２．変更年月日　２００１年　７月　２日
［変更理由］　住所変更
住　所　東京都港区芝浦一丁目１番１号
氏　名　株式会社東芝